BUY IN PRINT WORLDWIDE! magpi.cc/store

The MagPi

The official Raspberry Pi magazine | Issue 71 | July 2018 | raspberrypi.org/magpi

RUN ANDROID ON RASPBERRY PI

- Use Android Apps
- Build Touchscreen Devices
- Control Electronics

MAKE YOUR OWN OSCILLOSCOPE

Test voltage with Raspberry Pi & Arduino

MAGIC: THE GATHERING CARD COUNTER

Using AI to evaluate cards

BUILD A WILDLIFE CAMERA

Capture the critters in your garden

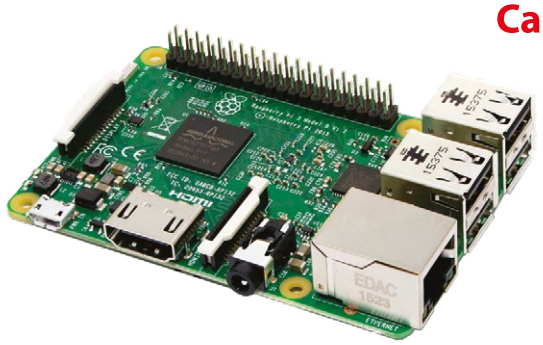
HERE COMES THE SUN

Track the weather with Raspberry Pi

ADD AI TO YOUR PROJECTS WITH TENSORFLOW

GLOBAL DELIVERY
magpi.cc/store
Raspberry Pi PRESS

CanaKit™



CanaKit Raspberry Pi 3 Ultimate Starter Kit

Model B | 1 GB RAM | 1.2 GHz | Quad-Core CPU

> Learn to Code
> Explore Computing
> Get started with Electronics

KIT INCLUDES RASPBERRY PI 3 AND ...

- PREMIUM CASE & HEAT SINKS

- 2.5A POWER ADAPTER

- 32 GB CLASS 10 MICROSD CARD

PRE-LOADED WITH OPERATING SYSTEM

- USB MICROSD CARD READER

- PREMIUM HDMI CABLE

- QUICK-START GUIDE

- GPIO TO BREADBOARD INTERFACE BOARD

- RIBBON CABLE

- FULL-SIZE BREADBOARD

- JUMPERS

MALE TO MALE & MALE TO FEMALE

- LEDS

- RESISTORS & PUSH-BUTTONS

Available for worldwide shipping at:

WWW.CANAKIT.COM

Raspberry Pi Zero W

Now available at CanaKit!

WiFi™ Bluetooth®

ELECTRONIC KITS • ELECTRONIC PARTS • RASPBERRY PI • ARDUINO

WELCOME TO THE OFFICIAL MAGAZINE

We've been waiting to go all in on Android for a long time. We've all used Google's mobile operating system, and the prospect of having a Raspberry Pi running Android is just too good to miss (page 16). The unofficial solution is a bit 'creative', but we think it's tons of fun.

Speaking of creativity: the UK weather has always been a bit bonkers, but lately it's gone full-blown bizarre. When it's not blazing hot, it's stormy, or chilly, and only a few months back we were in the depths of a bleak midwinter. We wouldn't be surprised by frogs at this rate.

So we are delighted to share Raspberry Pi's new Weather Station instructions (page 70). Building your own weather station is an amazing summer project, and the output is a perfect introduction to measuring data.

We've also got some extra-neat projects this month, including the Seeing Wand (page 30) and the Magic: The Gathering card analysis machine (page 34). Both use AI to push the Raspberry Pi further than we thought possible. If you are getting into AI, then don't forget to install TensorFlow on a Raspberry Pi (page 58).

There's a lot to get stuck into this month, so let's get started.

Lucy Hattersley
Editor

YOUR OFFICIAL RASPBERRY PI MAGAZINE
The MagPi
RUN ANDROID ON RASPBERRY PI
Use Android Apps
Build Touchscreen Devices
Control Electronics
MAKE YOUR OWN OSCILLOSCOPE
MAGIC: THE GATHERING CARD COUNTER
BUILD A WILDLIFE CAMERA
ADD AI TO YOUR PROJECTS WITH TENSORFLOW
BUY IN PRINT WORLDWIDE! magpi.cc/store
HERE COMES THE SUN

GET A PI ZERO W WITH A SUBSCRIPTION! PAGE 28

SUBSCRIBE FROM £5 SEE PAGE 28 FOR DETAILS

THIS MONTH:

16 ANDROID ON RASPBERRY PI
Get Google's OS up and running

44 BUILD AN OSCILLOSCOPE
Measure signal voltage with a Raspberry Pi

58 INSTALL TENSORFLOW
Get started with Google's AI framework

70 MAKE A WEATHER STATION
Come rain or shine, you'll know what's up

FIND US ONLINE raspberrypi.org/magpi **GET IN TOUCH** magpi@raspberrypi.org

The MagPi

Available on the App Store

GET IT ON Google Play

EDITORIAL
Editor: **Lucy Hattersley**
lucy@raspberrypi.org
Features Editor: **Rob Zwetsloot**
rob.zwetsloot@raspberrypi.org
Sub Editors: **Phil King and Jem Roberts**

DESIGN
Critical Media: **criticalmedia.co.uk**
Head of Design: **Dougal Matthews**
Designers: **Mike Kay and Lee Allen**
Illustrator: **Sam Alder**

PUBLISHING
For advertising & licensing:
Publishing Director: **Russell Barnes**
russell@raspberrypi.org | **+44 (0)7904 766523**
Director of Communications: **Liz Upton**
CEO: **Eben Upton**

DISTRIBUTION
Seymour Distribution Ltd
2 East Poultry Ave
London
EC1A 9PT | **+44 (0)207 429 4000**

SUBSCRIPTIONS
Raspberry Pi Press
Mann Enterprises, Unit E, Brocks Business Centre, Haverhill, CB9 8QP
magpi.cc/subscribe

CONTRIBUTORS
Alex Bate, Brian Beuken, Jody Carter, David Crookes, Richard Hayler, Nicola King, Nik Rawlinson, Matt Richardson, Richard Smedley, Mark Vanstone, Clive Webster

recycle
When you have finished with this magazine, please recycle it.

This magazine is printed on paper sourced from sustainable forests and the printer operates an environmental management system which has been assessed as conforming to ISO 14001.

The MagPi magazine is published by Raspberry Pi (Trading) Ltd., 30 Station Road, Cambridge, CB1 2JH. The publisher, editor, and contributors accept no responsibility in respect of any omissions or errors relating to goods, products or services referred to or advertised in the magazine. Except where otherwise noted, content in this magazine is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0). ISSN: 2051-9982.

Raspberry Pi PRESS

YOUR OFFICIAL RASPBERRY PI MAGAZINE
The MagPi
RUN ANDROID ON RASPBERRY PI
MAKE YOUR OWN OSCILLOSCOPE
MAGIC: THE GATHERING CARD COUNTER
BUILD A WILDLIFE CAMERA
HERE COMES THE SUN
ADD AI TO YOUR PROJECTS WITH TENSORFLOW

Contents

Issue 71 July 2018 raspberrypi.org/magpi

TUTORIALS

> PI 101: REMOTE ACCESS VIA VNC **42**
Connect to your Pi remotely using VNC

> BUILD AN OSCILLOSCOPE **44**
Use a Raspberry Pi and Arduino to make your own

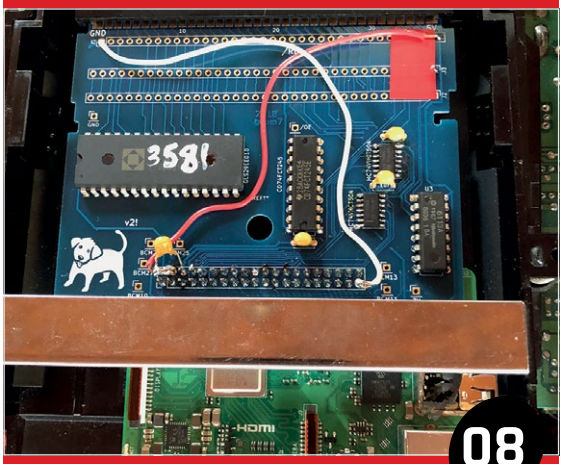
> LEARN PYGAME ZERO: PART 1 **52**
Making games is easy with Pygame Zero

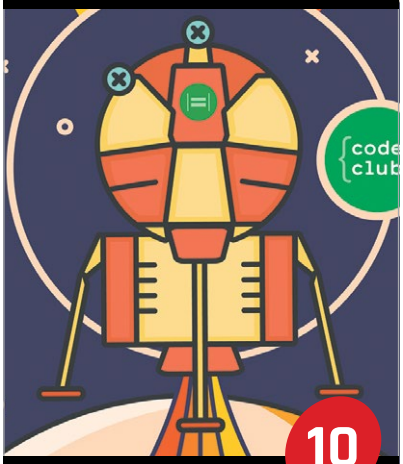
> USE AI WITH TENSORFLOW **58**
Google's AI functions work on Raspberry Pi

> MAKE GAMES IN C PART 7 **60**
Learn how to build maps with your game elements

> TAKE NATURE PHOTOS **64**
Build a wildlife trap to snap some critters

IN THE NEWS

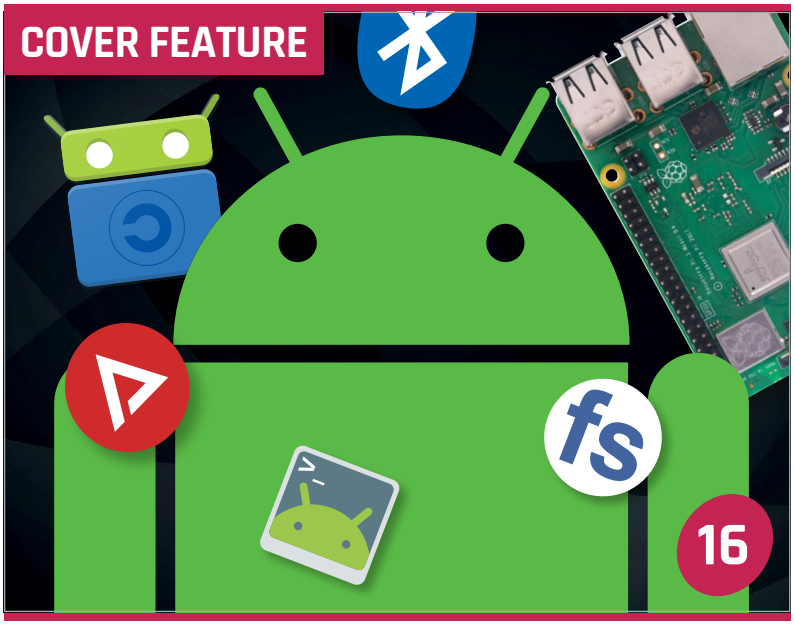
POWERED UP NES

08
Reverse-emulating an NES console

MOONHACK

10
Break another record!

PI GLIDER

12
A novel high-altitude craft

COVER FEATURE

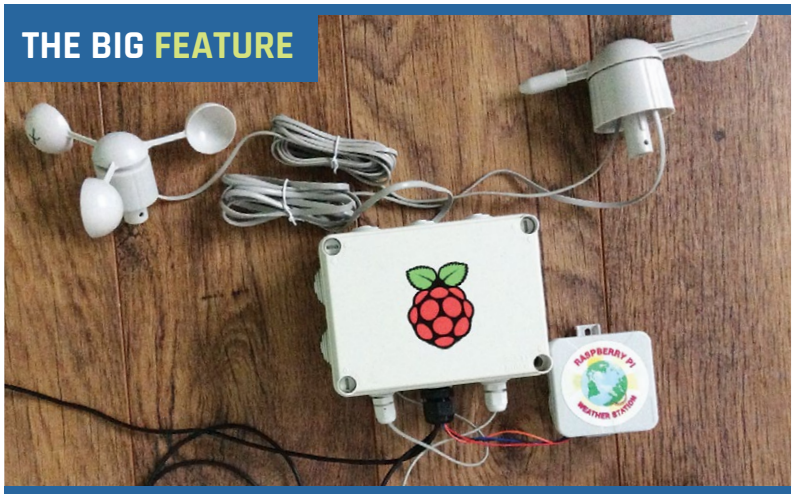

16

ANDROID ON PI

Some of the tools and techniques shown in The MagPi magazine are dangerous unless used with skill, experience, and appropriate personal protection equipment. While we attempt to guide the reader, ultimately you are responsible for your own safety and understanding the limits of yourself and your equipment. Children should be supervised. Raspberry Pi (Trading) Ltd does not accept responsibility for any injuries, damage to equipment, or costs incurred from projects, tutorials or suggestions in The MagPi magazine. Laws and regulations covering many of the topics in The MagPi magazine are different between countries, and are always subject to change. You are responsible for understanding the requirements in your jurisdiction and ensuring that you comply with them. Some manufacturers place limits on the use of their hardware which some projects or suggestions in The MagPi magazine may go beyond. It is your responsibility to understand the manufacturer's limits.

Contents

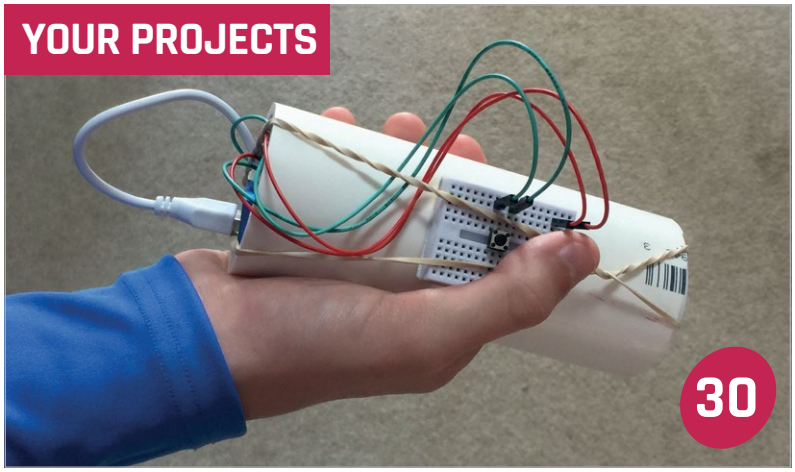
THE BIG FEATURE



WEATHER STATION 70

Build a Pi-powered weather station

YOUR PROJECTS

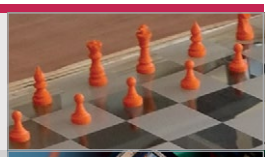


SEEING WAND

Using Microsoft's AI tools, it will tell you what's ahead of you

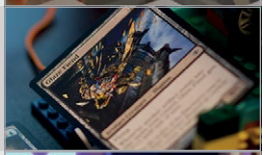
GHOST CHESS **32**

A spooky chess game with a haunted Pi



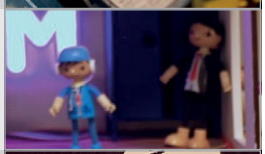
CARD APPRAISAL **34**

Find out how much your trading cards are worth



MYHOUSE **36**

An automated doll's house



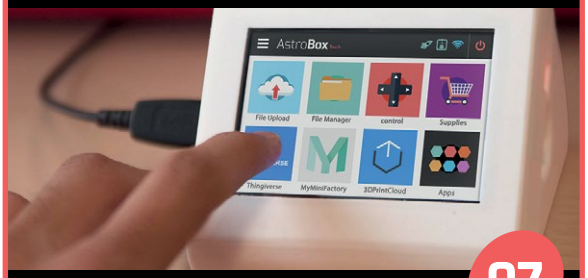
VOMIT COMIC ROBOT **38**

Print out randomly generated comics

WIN!

In association with Astroprint

ASTROBOX TOUCH & GATEWAY



We've got four AstroBox kits up for grabs to control your 3D printer from anywhere! 97

REGULARS

> NEWS **06**

> TECHNICAL FAQ **68**

> BOOK REVIEWS **84**

> FINAL WORD **98**

COMMUNITY

> THE MONTH IN RASPBERRY PI **86**

Translate some resources this month

> BRIAN CORTEIL PROFILE **90**

We talk to one of the top UK Pi robot makers

> EVENTS **92**

What events are happening this month

> YOUR LETTERS **94**

Excellent answers to wonderful questions

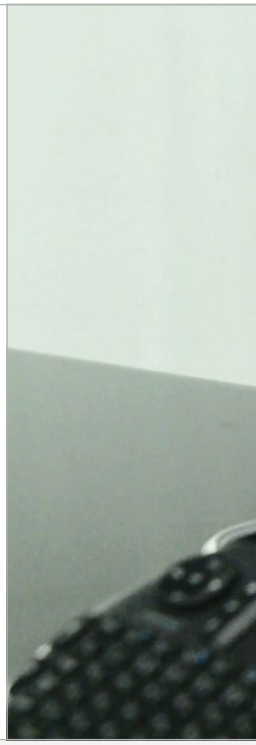
REVIEWS

> YETIBORG V2 **80**

> PI-HOLE **82**

CROWPI FLIES PAST ITS KICKSTARTER CAMPAIGN GOAL

The case for hacking and making



Chinese outfit Elecrow's latest project is a briefcase full of everything a hardware hacker and digital maker needs, from circuit boards and sensors to an integrated touchscreen display.

CrowPi crams a big bundle of electronics into its hard-shell carry case, and is compatible with all versions of Pi 2, Pi 3, and Pi Zero. The case comes in four colours: black, purple, blue, and yellow.

Among the fairly standard fare such as sensors for light, motion, and sound you will find various readout devices, an RFID sensor, an ultrasonic sensor, servo interfaces, a step motor connection, and a GPIO LED indicator.

This panel helps debug circuits, as Elecrow sales and marketing manager Jarvan explains, "The status LED indicates when a sensor if working and how, [which] gives you a better understanding of how microelectronics work."

Elecrow worked hard to "choose the most common and project-orientated sensors" to include, Jarvan reveals, but had to "make sure we can fit as many sensors as possible" into the Elecrow case.

Jarvan adds, "We've tested a lot of LCD screens [to] choose the most suitable one, and even left some space [above] for the Raspberry Pi camera."

Below Plug in the power bank accessory and you can use CrowPi like a laptop or portable hacking station

Crow funding

Jarvan tells us that "the Elecrow philosophy is 'make your making easier'", and this belief isn't just represented in how CrowPi brings together so much hardware, but in its documentation too.

"We've been working very hard writing those tutorials," says Jarvan, modifying the documentation "hundreds of times in order to be as simple and understandable as possible." This makes CrowPi as useful for schools and clubs as for individuals wanting a neat kit to develop code or hardware, according to Jarvan.

CrowPi runs a standard installation of Raspbian, with only

the "basic drivers [required] in order for our hardware to operate right out of the box," Jarvan confirms. "We wanted to modify the Raspbian image as little as possible, to keep the right for the customer to choose what they want."

Elecrow intends to sell CrowPi units through its website, **elecrow.com**, and Jarvan confirms a possibility "to sell it on our Amazon or AliExpress store."

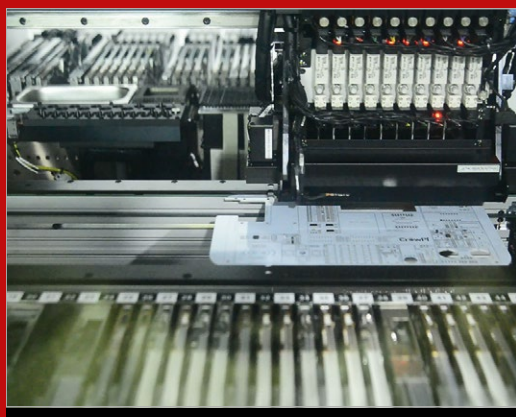
For more details, visit the CrowPi Kickstarter page: **magpi.cc/oUFbTt**.

> The Elecrow philosophy is 'make your making easier'

A basic, Pi-less unit costs $229 (£170), while the $369 (£275) Advanced Kit includes a Pi 3B+, a built-in camera, two gamepads, and a wireless keyboard and mouse, plus a few other hardware configurations to select. As Elecrow has its own production line, Jarvan says "delivery time and quality is guaranteed."

Below CrowPi comes in four colours and a few different hardware configurations, so you can choose which suits you best

THE MAKING OF CROWPI



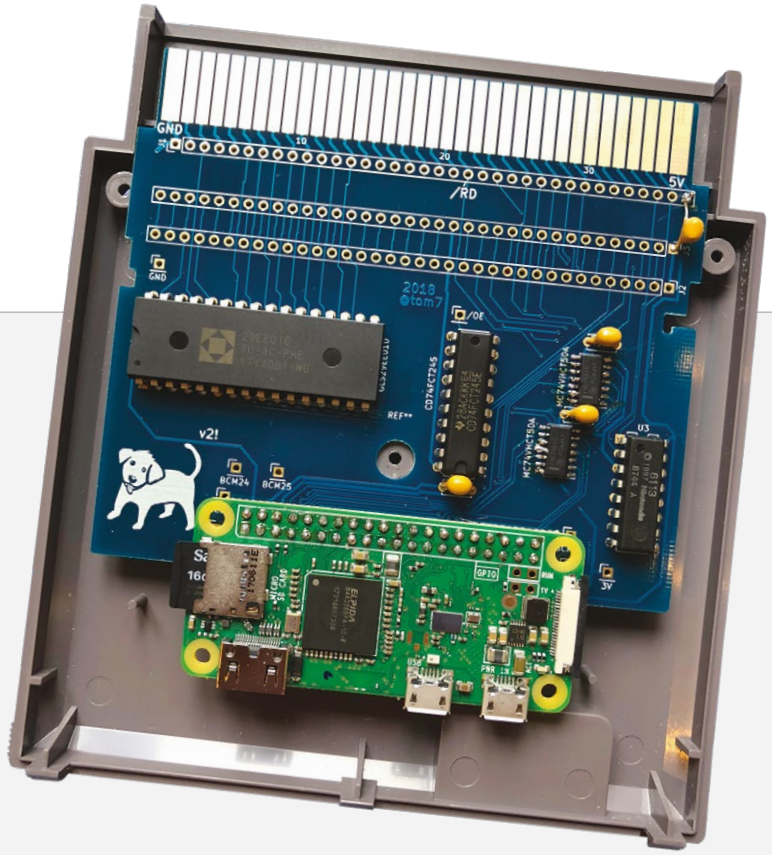
Elecrow is based in Shenzhen City, China, the electronics capital of the world. Sales and marketing manager Jarvan was keen to point out the advantages of Elecrow's production line and gave us some glimpses of the process.

You can see the RFID contact pad to the lower left of the CrowPi circuit board. RFID can be a fun and useful way to trigger events using an Android phone, or newer model iPhone.

The buttons can be programmed to do whatever you like – the cluster might be used as a keypad while the four to the left could control a character in a self-made game.

'REVERSE-EMULATED' NES

Pi plays SNES games on unmodified NES



Right The project required a bespoke circuit board, housing the Raspberry Pi, a reprogrammed NES PPU chip, and a Nintendo CIC content protection chip

Emulation is the process of running older software on newer, incompatible hardware, but Tom Murphy VII, a computer scientist from Pittsburgh, PA, wanted to turn that process on its head by 'reverse-emulating' a SNES.

The basic idea was to run SNES games on a NES, to tell a joke based on what Tom calls 'improper hierarchy'. It's not funny to run old software on new hardware, because software is meant to get better over time and older hardware is meant to become obsolete. However, Tom contends that it is funny to run newer software on older hardware, because that's just an absurd thing to do, or an example of 'improper hierarchy'.

The crux of Tom's reverse-emulation was to somehow fool an unmodified NES into running the SNES launch title Super Mario World, which he achieved by embedding a Raspberry Pi 3 into a NES game cartridge. The Pi emulated the SNES game, but then had to convert the graphics into a form that the NES hardware could display.

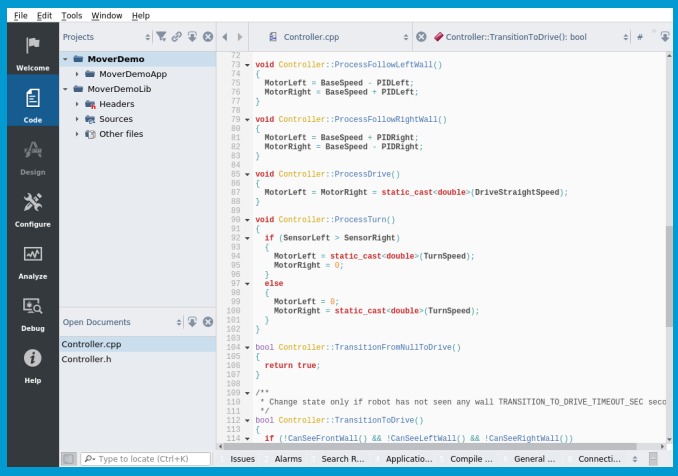
As Tom clarifies for us, the Pi is acting "kind of like a downsampler, although the SNES pixel resolution is actually the same as the NES, so it's not downsampling in that spatial sense." Instead, the Pi has to convert the larger numbers of colours in the SNES game into the limited colour palette of the NES. "I prefer to think of it like an absurdly advanced 'memory mapper' within the cartridge," Tom tells us.

> "I prefer to think of it like an absurdly advanced 'memory mapper' within the cartridge"

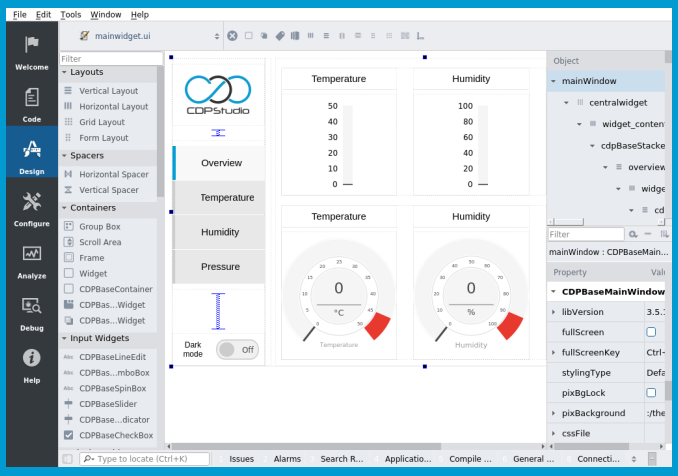
While Tom admits that he might have had an easier build by using a bespoke microcontroller, he reveals that he used a Raspberry Pi so "I can have my whole development environment right on the device." Rather than "swapping chips into a programmer, or even attaching an ICSP cable," Tom continues, the Pi "really shortens the edit/compile/test cycle."

Tom explains the many technical issues he had to solve in his fascinating video at **youtu.be/ar9WRwCiSro**.

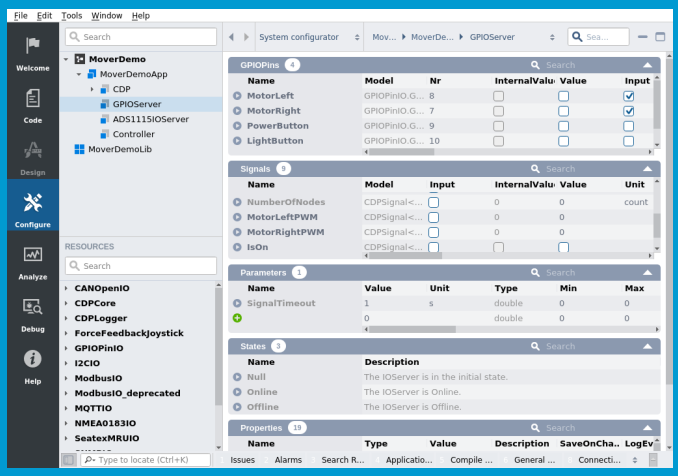
Below Tom originally used a Pi Zero W, but found it a little slow for the task so upgraded to a Pi 3 for the final build



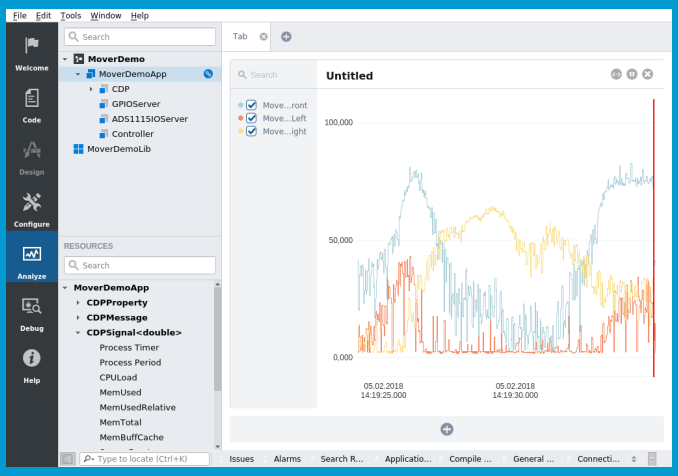
Code



Design



Configure



Analyze

Now free for home projects

A professional control system development tool

CDP Studio is a development platform for industrial control systems, now coming with a free version for non-commercial use. The system can run on a Raspberry Pi, supports C++, open source libraries and has a large feature toolbox including GPIO, I2C and MQTT. Its built in GUI design tool and features lets you code less and do more.

Free download on www.cdpstudio.com

CDP Technologies AS
Nedre Strandgate 29
P.O. Box 144
NO-6001 Ålesund, Norway

Tel: +47 990 80 900
info@cdptech.com
www.cdpstudio.com

CDPStudio

MOONHACK 2018

Help break a coding world record

This year's Moonhack will be even bigger than before, with an estimated 50 000 kids completing challenges and running projects to explore space on 20 July.

As Nicola Curnow, national programme manager at Code Club Australia reveals, "We're setting a world record, inviting every kid in the world to give coding a go, or impress us with their skills. [...] Last year we had 28 575 kids from more than 50 countries participating; we're aiming for 50 000 kids this year."

You can register for Moonhack at **moonhack.com** to get your school, library, Code Club, CoderDojo, or your family involved, but the nature of the projects is still "top secret stuff!" says Nicola. However, she tells us "The whole Moonhack campaign is designed to be super-accessible, so there is no hardware required to participate."

Moonhack projects are all space-themed, as the event celebrates the Apollo 11 moon landing. Stay informed via #moonhack.

Left Join the anticipated 50 000 Moonhackers worldwide at moonhack.com

MODULAR MAKING WITH TIBBO-PI

A LEGO kit for electronics

Above Connect the modular Tibbits to the mainboard to prototype your hardware as easily as building a simple LEGO model

Tibbo Technology's Tibbo Project System (TPS) has been around for four years, but it's finally coming to the Raspberry Pi.

Rather than using breadboards or soldering, the TPS uses Tibbit blocks to "implement an I/O function" by plugging them into a TPS 'mainboard' (motherboard). Announced at this year's Computex technology show in Taiwan, there will soon be a mainboard that incorporates a Raspberry Pi 3.

It is unclear whether the Pi 3B+ will be supported, but with 60 Tibbit blocks to add anything from power relays to audio converters to real-world sensors, you should be able to construct almost any project as quickly as a basic LEGO model.

Details are still in the process of translation, but Tibbo confirms that Node-RED (JavaScript) is the standard coding language for Tibbo-Pi, and that C and Python are alternatives.

An English brochure for Tibbo-Pi is planned, available through **co-works.jp/tibbo-pi**, but this site is in Japanese. However, Tibbo's main site lists all the currently available Tibbits in English – see **tibbo.com**.

ANDROID NOUGAT ON A PI

Major update to RaspAnd brings Bluetooth and better wireless

A new release of the RaspAnd operating system brings Android Nougat to the Raspberry Pi 3B, preloaded with apps such as TeamViewer, Kodi, and Aptoide TV.

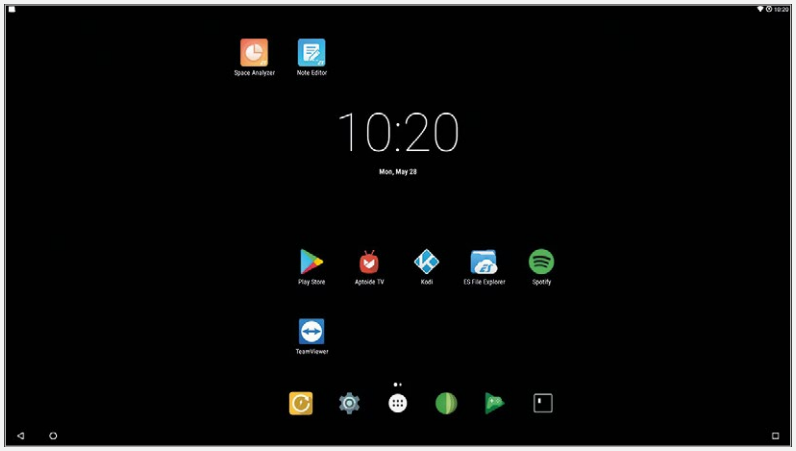
RaspAnd is developed by Swedish outfit Exton Linux (**exton.se**), with modifications made to allow better support and compatibility with standard Raspberry Pi hardware. Alas, only the Pi 3B is supported at this time – the Pi 3B+ is not.

> "RaspAnd Nougat 7.1.2 incorporates Bluetooth"

Crucially, RaspAnd Nougat 7.1.2 incorporates Bluetooth support for the first time, as well as a 'stable' wireless connection which automatically reconnects after every reboot.

RaspAnd includes Google Play Services and the Play Store, meaning you have full access to any app hosted on the main store. This includes the recommended version 4.6 of Spotify and version 18.0-Alpha2 of Kodi.

To install RaspAnd, you need to download the image from **magpi.cc/SALgip**, which costs $9, and burn it to a microSD card using whichever method you prefer. Then follow the advice on RaspAnd's information page: **magpi.cc/ATERWF**.



Above RaspAnd is an easy way to deploy Google's Android OS on your Raspberry Pi 3B

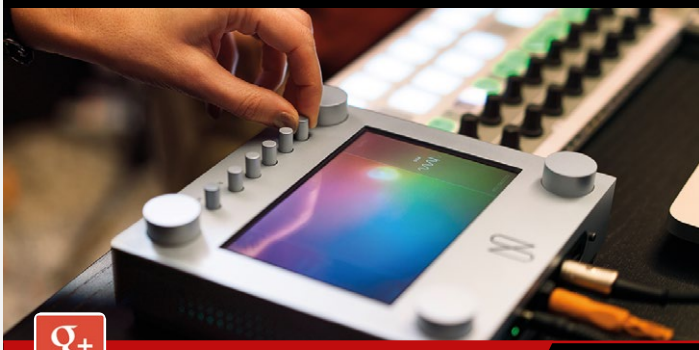
NOW TRENDING

The stories we shared that flew around the world

TOUCHSCREEN CONTROL AND DATA SYSTEM

magpi.cc/dUGpoR

Peter Juett's project uses standard components to show information such as weather and air quality, and allows you to control smart lights and other IoT devices such as WeMo smart plugs.



GOOGLE NSYNTH SUPER

magpi.cc/gPUAnS

Google's NSynth system uses AI learning algorithms to create new sounds based on samples of musical instruments. The NSynth Super houses this next-gen synth in a touchscreen device you can build yourself.

JOIN US AT RASPBERRY FIELDS

magpi.cc/xwwGPL

Come to the first annual festival of technology and science, held at Cambridge Junction over the weekend of 30 June and 1 July. Highlights include brewing a magic potion and Museum in a Box.

HIGH-ALTITUDE RASPBERRY PI GLIDER

Effortless HAB data recovery

Left Izzy's glider isn't anything fancy – mainly foam board and corrugated plastic, with handwarmers to keep the electronics inside from freezing

Above The day of the launch, using a $60 military surplus sounding balloon to send the glider into the atmosphere

Izzy Brand, a student at Brown University, USA, created a clever system to recover the data from his high-altitude balloon (HAB). Rather than use a parachute and geolocator, he instead fitted the data module to a glider set to land at preprogrammed co-ordinates.

Izzy reveals, "I had the idea a long time ago – maybe early high school (2013)." Initially the idea was just to find a way to fly a glider, dropping it from a hot-air balloon, as Izzy's "nearby hills weren't steep enough."

The glider uses a Raspberry Zero W and a Pixhawk, a flight controller powered by an ARM processor. "I chose the Zero W," Izzy explains, "because it can run MAVProxy, essentially a terminal version of the GUI-based ground station software used to control the Pixhawk." Izzy chose the Pixhawk due to his familiarity with its predecessor, the ArduPilot.

At 10 000 m, the Zero W turned on the autopilot mode and "triggered a solid-state relay to burn the nickel-chromium wire and release the glider."

Guided landing

Izzy explains that the Pixhawk module's autopilot mode operates on a system of waypoints, so he set "only one waypoint co-ordinate at the target landing location, with an elevation of zero."

> "I had the idea a long time ago – maybe early high school (2013)"

In addition, the Pixhawk doesn't have a glider mode, so Izzy had to "set the maximum ascent angle to zero so the glider wouldn't try to climb without a motor and thereby stall."

Amazingly, after launching the balloon and driving to the landing site 122 miles away, the glider was waiting just 10 m from the target location.

"We were astonished," Izzy tells us. "This project failed miserably in 2015, [as] the glider landed in a forest about ten miles from its target."

Izzy would like to thank his friends Luke Fisher and Nick Menz for their help "in testing the glider and on the launch day." You can find the source code and flight logs at **magpi.cc/qkqKXj**.

NEW COMPUTER SCIENCE COURSE FOR IRISH SECONDARY SCHOOLS

Computer Science Teachers' Association of Ireland helps schools prepare

Below The CSTAI collaborates with other organisations, including CoderDojo; CSTAI President Stephen Murphy meets CoderDojo's Tara McGrath and the team from the Cork Institute of Technology CoderDojo

The third element of the Irish government's computer science curriculum will be trialled in September, and the Computer Science Teachers' Association of Ireland (CSTAI) is ready with resources to help teach the new Leaving Certificate, Computer Science course.

Following the previous Junior Certificate short courses in Coding and Digital Media Literacy, the new course aims to "develop an appreciation of the diverse roles of computing technology in society and the environment in which [students] live," according to CSTAI President Stephen Murphy.

The course specification outlines that "embedded systems (such as the Raspberry Pi) will play a central role in the practical component of this subject," explains Stephen.

He tells us that "over 200 schools applied for the Leaving Certificate, Computer Science pilot scheme," of which 40 were selected. The Leaving Certificate, Computer Science "will open as an elective subject for all schools to offer if they so wish" in September 2020.

Computer Science Teachers' Association of Ireland
Ríomheol Oidí na hÉireann

Above The CSTAI supports teachers of computer science, coding, and digital media literacy

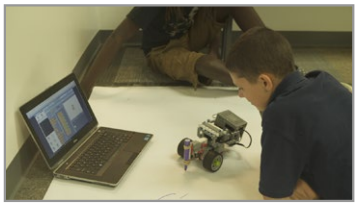
Supporting teachers

While the Irish government has been described as providing "excellent" support for teachers wishing to teach any of the three courses, the CSTAI supplements that with "hundreds of teaching resources (PowerPoints, worksheets, videos, etc.) for coding, computer science, digital media, and general computing courses," Stephen reveals.

Access to these resources is free for CSTAI members; to become a member, visit **magpi.cc/dTiylU** or email **president.cstai@gmail.com**.

The CSTAI was founded in November 2017 and has "over 440 members from Ireland, UK, New Zealand, and Dubai," says Stephen.

DEXTER INDUSTRIES
GoPiGo
20+ new projects for your Pi robot car.

BrickPi3
LEGO MINDSTORMS + RASPBERRY PI

www.dexterindustries.com

pi-top

Inspiring inventors and creators to seek the skills of tomorrow and create their future, today.

The modular laptop with sliding keyboard

8HR BATTERY LIFE

14" FULL HD 1080P SCREEN

180° HINGE

CUSTOM PASSIVE COOLING BRIDGE

MODULAR RAIL

pi-top

Colors

Raspberry Pi 3 optional

AWESOME INVENTOR'S KIT INCLUDED

20+ projects to explore

Explore beyond the screen and keyboard by creating with the all-new **pi-top** modular laptop.

Get started with 20+ inventions in the inventor's guide booklet. There are 3 inventor's journeys - Smart Robot, Music Maker and Space Race.

The modular desktop

14" FULL HD 1080P SCREEN

MODULAR RAIL

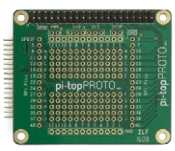
ADJUSTABLE VIEWING ANGLES

pi-topCEED

Colors

Raspberry Pi 3 optional

pi-topCEED is the plug & play modular desktop. It's the easiest way to use your Raspberry Pi. We've put what you love about our flagship laptop in a slimmer form factor. Join hundreds of code clubs and classrooms using **pi-top**CEED as their solution to Computer Science and STEAM-based learning.



pi-topPROTO

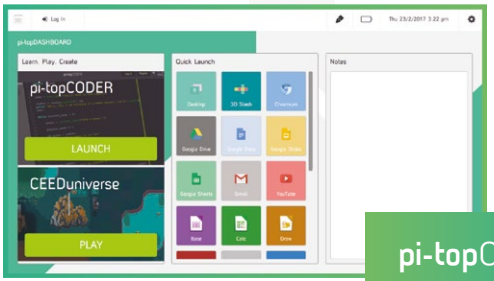
pi-topSPEAKER

pi-topPULSE

Modular **Accessories**

pi-top is an award-winning ecosystem designed to make experimenting, coding and building electronics, simple, affordable and fun. **pi-top**OS is here to guide you through the world of making!

The OCR* endorsed **pi-top**OS (Operating system) platform comes pre-installed on the SD card shipped with every unit. **pi-top**OS software suite lets you - browse the web, - check emails, - create and edit Microsoft Office compatible files. Gain access to dozens of hands-on learning lesson plans with **pi-top**CODER and have fun learning to code with CEED**universe**!



pi-topOS

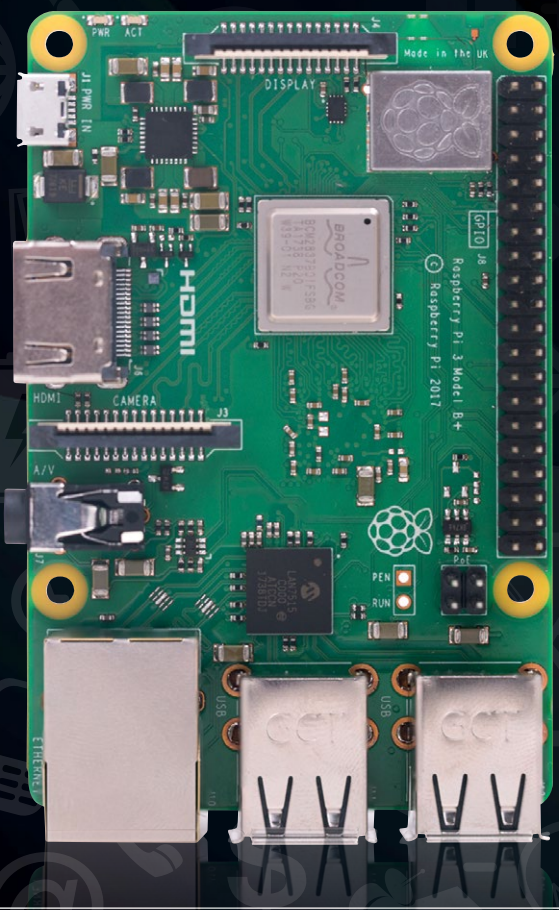
pi-topCODER has a fully integrated coding environment letting you program hardware, code in Python and learn lots of STEAM skills! Our integrated test framework gives you the ability to assess your own understanding as you learn.

CEED**universe**

Learn programming concepts through our minigames, for example, learn problem decomposition by solving visual programming puzzles.

RUN ANDROID ON RASPBERRY PI

How to get Google's Android operating system working on your Pi!



The Raspberry Pi has been used in many different ways over the years by many different people. Installing Android on Raspberry Pi, while a pretty popular idea, has never been super-easy, though. Solutions have existed in the past, but have never worked like true Android.

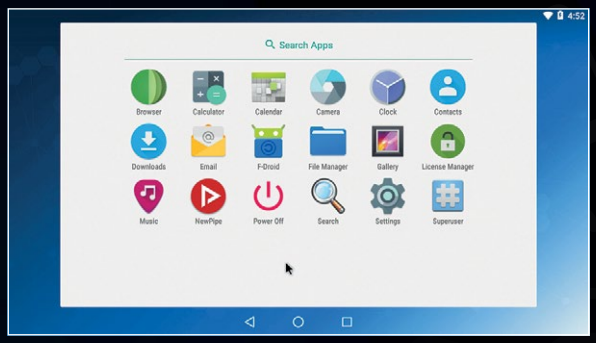
Recently that's all changed. While limitations stopping Android working properly on Raspberry Pi were lifted some time ago, it's taken until now for a build of full Android to be created by budding hackers to run on the Pi. We look at emteria.OS, the first proper Android release for Raspberry Pi.

That's not the only way to use Android on Raspberry Pi either, as Google has released an IoT-focused version called Android Things.

Not sure which one to use? We'll show you how to use both, and why you might want to, over the following pages...

CHOOSE YOUR ANDROID

Things or emteria.OS? You decide!



emteria.OS

emteria.com

Emteria.OS is a full build of Android available for Raspberry Pi. While you can get it for free and use it as an individual, it's aimed more towards industry. Embedded Android devices are big business, and marrying the mobile OS with the Pi opens up a whole new world of Android in both maker projects and consumer products.

PAGE 18

android things

Android Things

magpi.cc/bSHeqc

This stripped-down, official release of Android is perfect for IoT applications. It's a lot more simplistic than full Android, and you need to program it from a separate computer. However, we've seen people do some amazing things with it – see some of them on page 26.

PAGE 22

INSTALL EMTERIA.OS

Installing Android onto your Raspberry Pi made easy

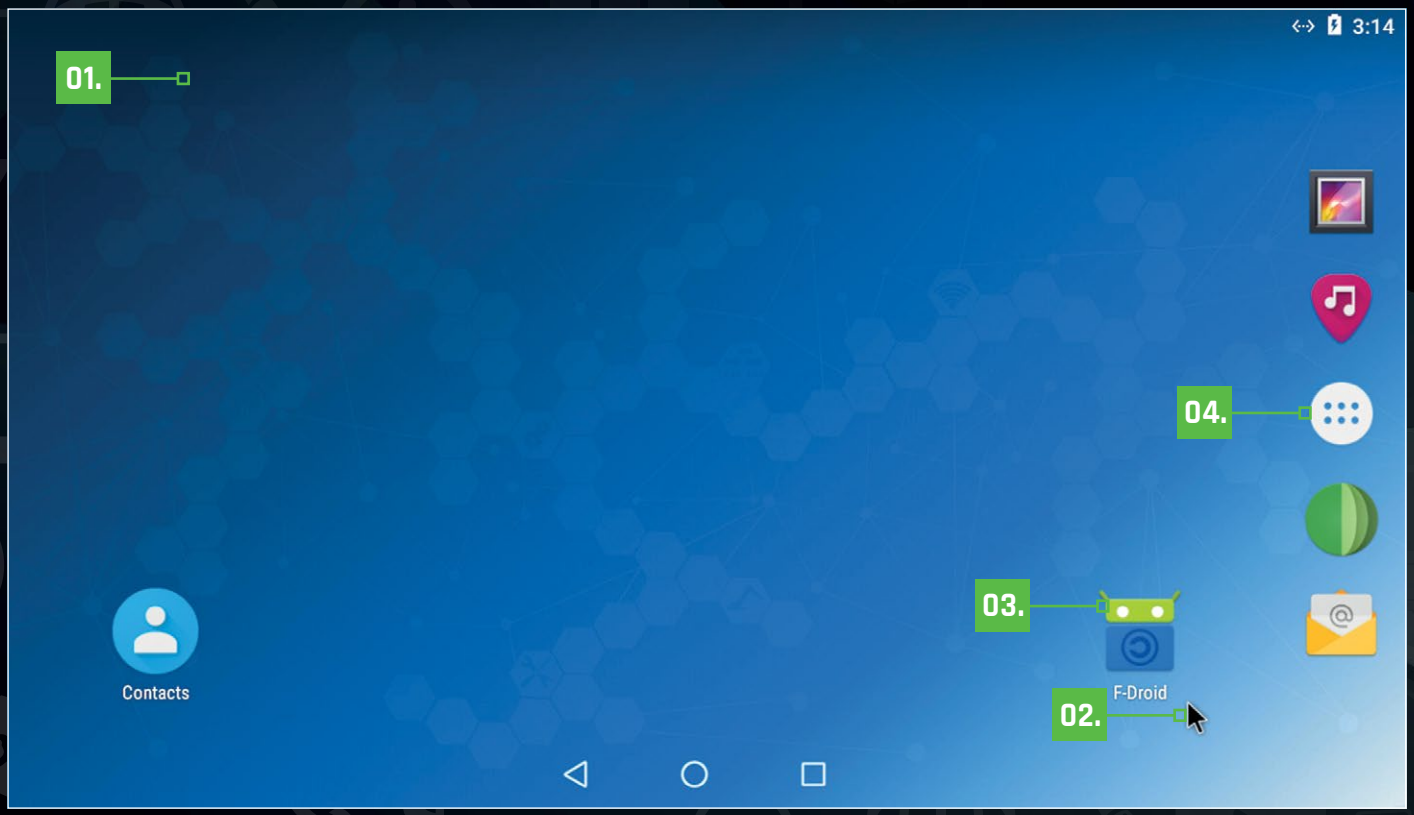
It's been a long time coming, but finally there's a proper way to install full Android on your Raspberry Pi thanks to emteria.OS. Based on RTAndroid, this stable version is a bit more aimed towards business; however, more personal-use versions will be coming soon. Here's how to set it up.

01. Desktop
The main desktop will be familiar to those that have used Android as a tablet device.

02. Mouse control
While emteria.OS does work with a number of Raspberry Pi touchscreens, you can still use it with a mouse and keyboard.

03. F-Droid
F-Droid is an app store alternative available on emteria.OS, allowing you to install a number of free apps.

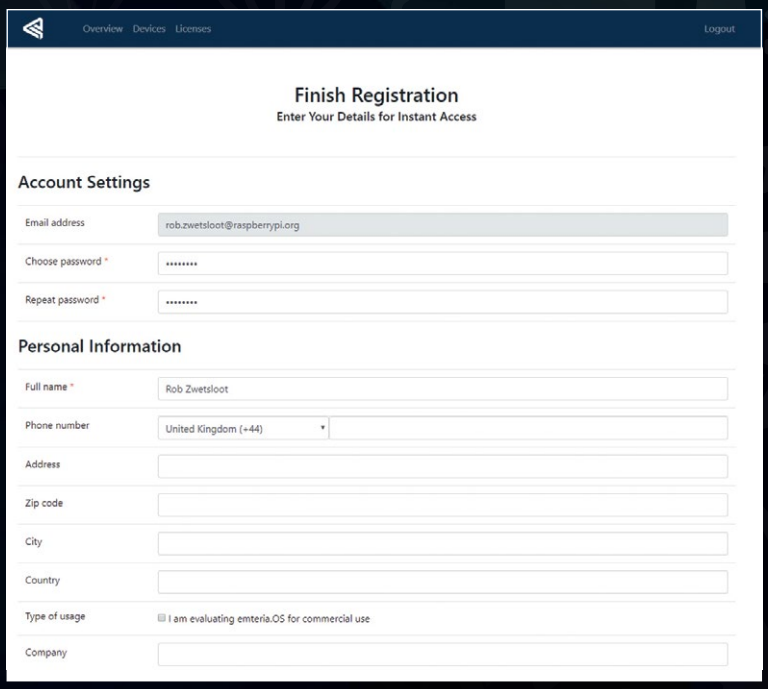
04. App drawer
All your other apps live in here and, like on other Android devices, you can move them onto your main display.



Register!

STEP 01

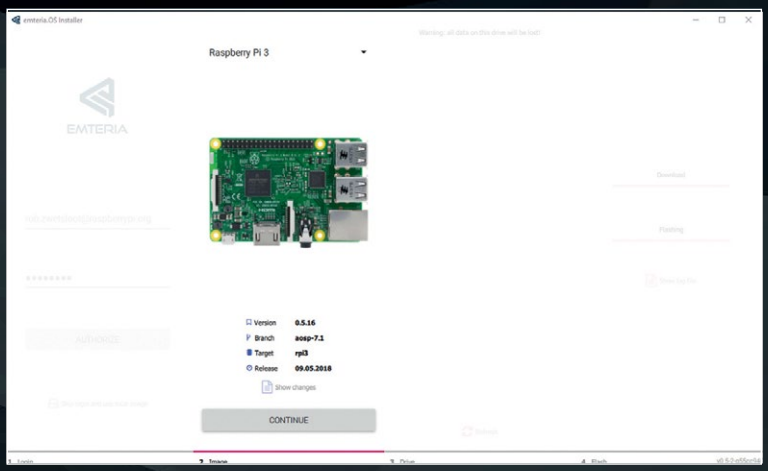
You'll need to first register an account on **emteria.com** – look for the Register option on the top right of the home page. Fill out the relevant details, confirm your email address, and then make sure you're logged into your account on the emteria.OS website.

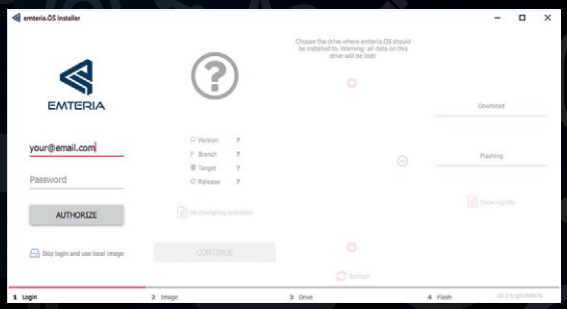


Flash the SD

STEP 03

All you need to do now is open the installer, enter your emteria.OS username and password, and select the Raspberry Pi 3 as your device of choice for installing. Next, you'll need to select your microSD card as the installation location and then just wait – the installer will download the image and then burn it to your microSD card. Now you're ready!





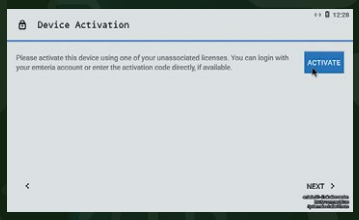
Get the installer

STEP 02

From your emteria.com account you can download the installer for your system. This will burn the image for emteria.OS onto a microSD card to be used in a Raspberry Pi. On Windows you'll have to install the installer using the setup wizard, while on macOS you'll have to drag it into the Applications folder and run it using superuser in the Terminal.

Emteria.OS prices

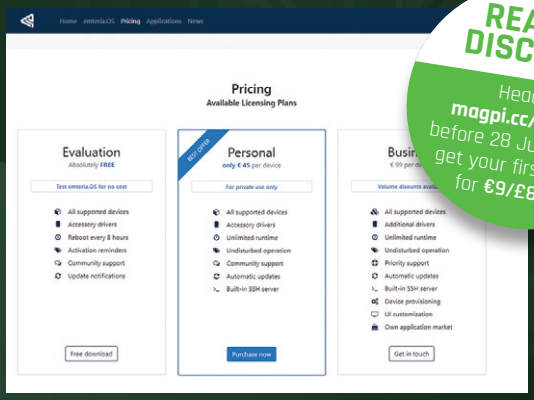
Free to test



During the initial setup of emteria.OS, you'll be asked to activate your device with a licence – clicking Next allows you to continue on with emteria.OS and use it for free. There are some limitations, however: it only stays on for eight hours at a time, and has a watermark in the corner. Perfect for testing it out with a few projects, though!

Purchasing a licence

There are full licences for purchase on emteria.com, including a cheaper 'personal' option and one suited for business that includes bulk-buy discounts.



READER DISCOUNT

Head to **magpi.cc/emteria** before 28 July 2018 to get your first month for **€9/£8/$10**!

INSTALLING APPS

Here are some extra ways to install software onto your emteria.OS-powered Pi

TOP F-DROID APPS

NewPipe

Play YouTube videos on your device with this small, lightweight app that doesn't require the original YouTube app to work.

Terminal Emulator

As emteria.OS is a bit more hackable than stock Android, you'll be able to make good use of this terminal emulator to run more advanced commands.

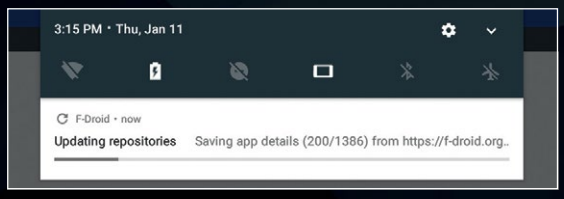
Face Slim

Missing out on some social media time? Face Slim allows you to use Facebook much like the standard app.

INSTALLING F-DROID APPS

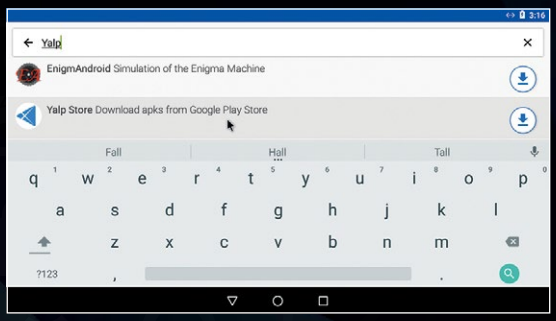
Start F-Droid — STEP 01

Find F-Droid on your home screen on in the apps menu and start it up. It won't launch straight away as it will first need to update all its repositories – something you'll be familiar with from installing software on Raspbian using apt-get. This might also take a while so please be patient.



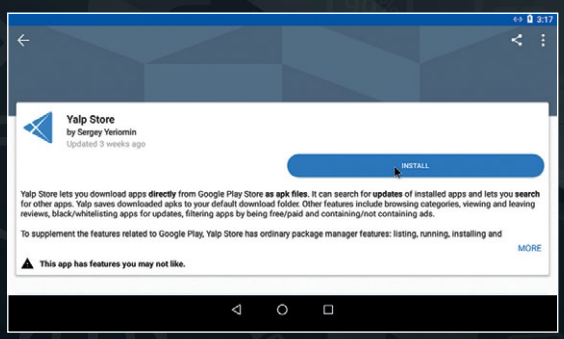
Find an app — STEP 02

F-Droid works very similarly to the usual Play Store on normal Android devices. There are categories you can look for apps in, and you can also search for something you like. Some of your favourite apps on Android may even be available.



Install! — STEP 03

Select the app you want, then hit the install button. It's as easy as that – F-Droid even includes some info on the apps as well. F-Droid also lets you keep the apps up-to-date, although it's a bit more manual than the Play Store equivalent.



SIDELOADING

What is sideloading?

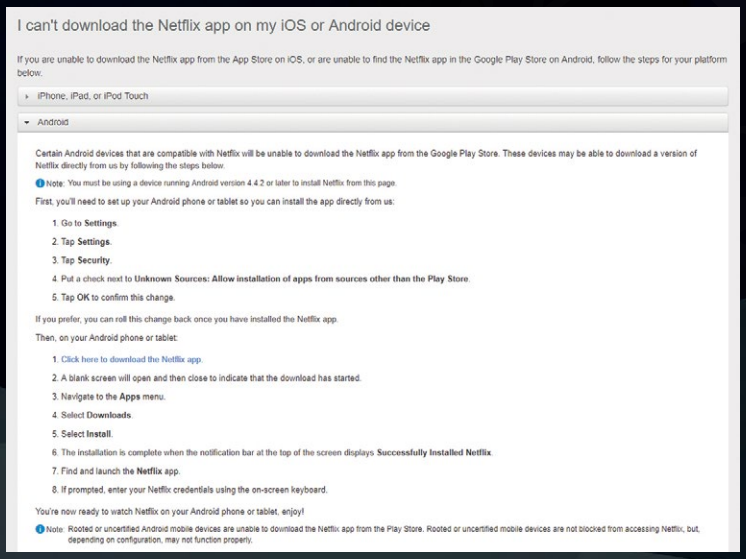
STEP 01

Android apps work by downloading a specific file which is then used to install the software onto the device. These files are called APKs, as that's the file name extension (.apk). If you have the APK file, you can install it on Android without having to go through a separate store.

Example APK

STEP 02

F-Droid specialises in free and open-source Android apps, so you may not find everything you want on there, such as Netflix. As Netflix is more concerned about you being able to watch the shows you're already paying for, the APK for the app is available from its website. Open the browser in emteria.OS and head to **magpi.cc/ZUBOSL**.



I can't download the Netflix app on my iOS or Android device

If you are unable to download the Netflix app from the App Store on iOS, or are unable to find the Netflix app in the Google Play Store on Android, follow the steps for your platform below.

▸ iPhone, iPad, or iPod Touch

▾ Android

Certain Android devices that are compatible with Netflix will be unable to download the Netflix app from the Google Play Store. These devices may be able to download a version of Netflix directly from us by following the steps below.

Note: You must be using a device running Android version 4.4.2 or later to install Netflix from this page.

First, you'll need to set up your Android phone or tablet so you can install the app directly from us:

1. Go to **Settings**.
2. Tap **Settings**.
3. Tap **Security**.
4. Put a check next to **Unknown Sources: Allow installation of apps from sources other than the Play Store**.
5. Tap **OK** to confirm this change.

If you prefer, you can roll this change back once you have installed the Netflix app.

Then, on your Android phone or tablet:

1. Click here to download the Netflix app.
2. A blank screen will open and then close to indicate that the download has started.
3. Navigate to the **Apps** menu.
4. Select **Downloads**.
5. Select **Install**.
6. The installation is complete when the notification bar at the top of the screen displays **Successfully Installed Netflix**.
7. Find and launch the **Netflix** app.
8. If prompted, enter your Netflix credentials using the on-screen keyboard.

You're now ready to watch Netflix on your Android phone or tablet, enjoy!

Note: Rooted or uncertified Android mobile devices are unable to download the Netflix app from the Play Store. Rooted or uncertified mobile devices are not blocked from accessing Netflix, but, depending on configuration, may not function properly.

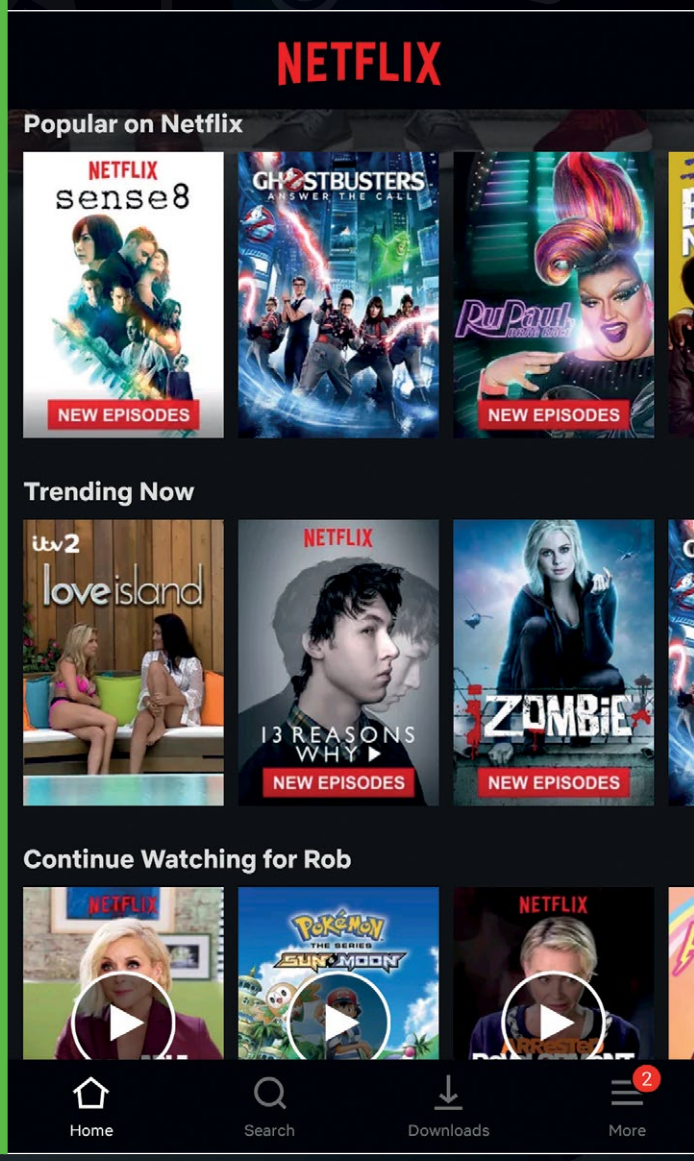
Using the Google Play Store

With custom versions of Android, there are some restrictions. One of the big ones is that the Google Play Store and some of the proprietary Google apps are not allowed to be included with software. However, this doesn't stop you from installing APKs using sideloading.

Run the APK

STEP 03

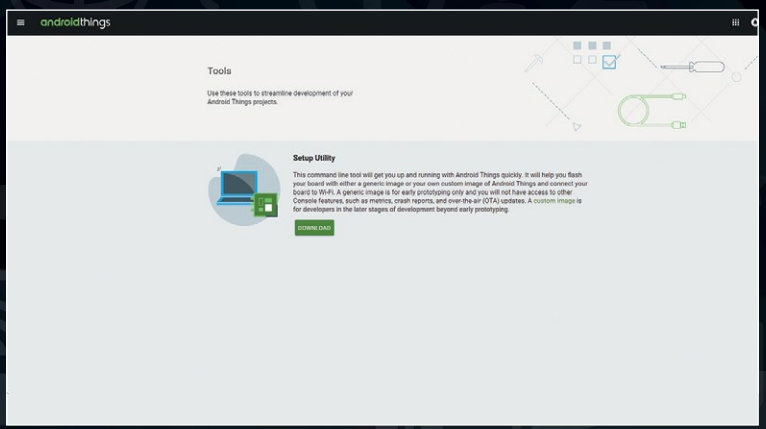
Once the APK file has been downloaded, open it from the Downloads menu. You may need to enable installing from unknown sources if you've turned that option off in the emteria.OS settings – otherwise it will go about installing the Netflix APK for you. It will appear in your app menu as usual, although you'll need to perform manual updates every now and then!



INSTALL ANDROID THINGS

You too can do IoT with Android Things and a Raspberry Pi

A simpler way of putting Android on the Raspberry Pi, Android Things is optimised more for IoT. You won't be getting your home screen or app menu or anything like that, but you can program it to be a bit more customisable than stock Android.


androidthings
Tools
Use these tools to streamline development of your Android Things projects.
Setup Utility
This command line tool will get you up and running with Android Things quickly. It will help you flash your board with either a generic image or your own custom image of Android Things and connect your board to Wi-Fi. A generic image is for early prototyping only and you will not have access to other Console features, such as metrics, crash reports, and over-the-air (OTA) updates. A custom image is for developers in the later stages of development beyond early prototyping.
DOWNLOAD

Get the installer

STEP 01

Head to the Android Things console (**magpi.cc/iXejLF**) and sign in with your Google account. Click on the menu in the top left to find the Tools section, where the Android Things Setup Utility lives. You'll need to download this and then unzip the files within – there's no extra installation for the setup utility.

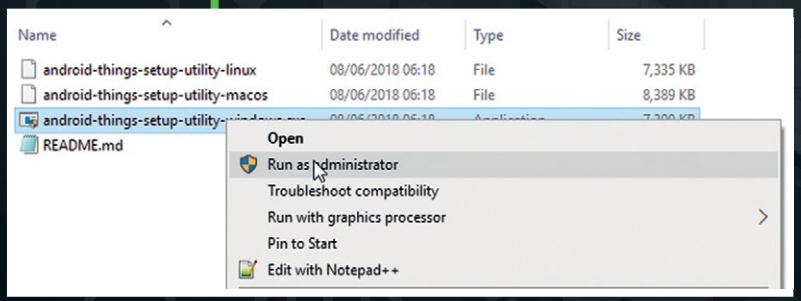
Start it up

STEP 02

On Windows, you'll need to right-click on the Setup Utility EXE (which has 'windows' in the file name) and run it as Administrator. On macOS and Linux, you'll need to run it from a terminal with something like:

```
sudo ~/Downloads/android-things-setup-utility/
android-things-setup-utility-linux
```

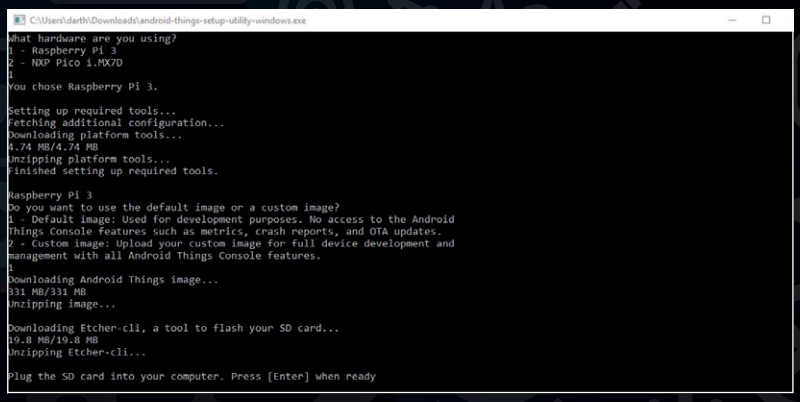
...but making sure to point towards the correct file in the correct folder.

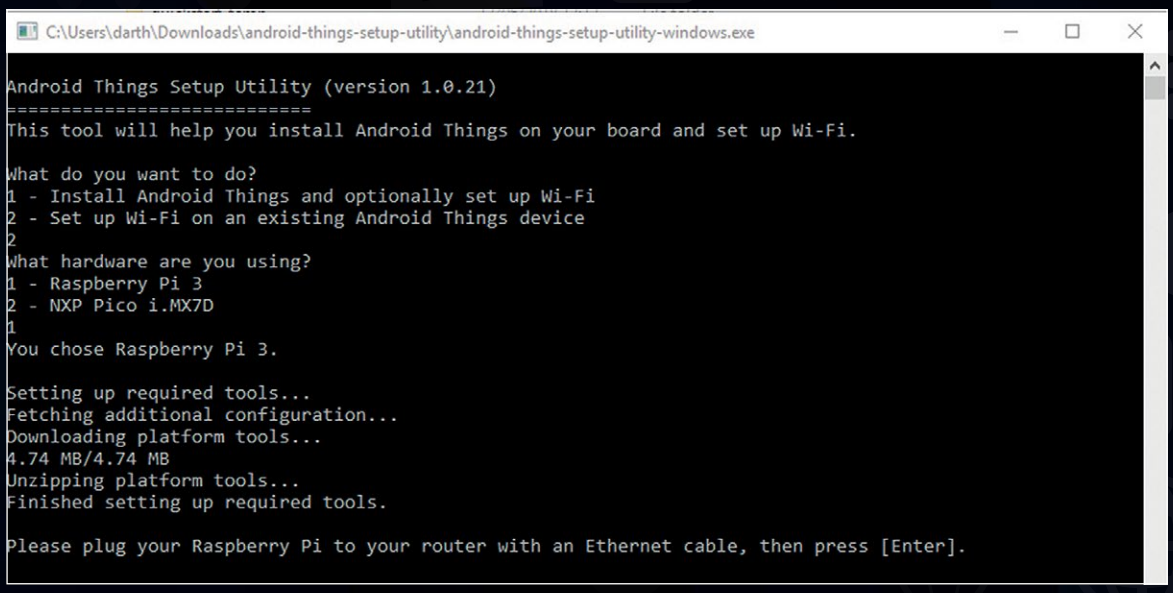


Flash the SD card

STEP 03

A command prompt window will appear with two options – you'll want to choose 'Install Android Things and optionally set up WiFi' by typing **1** and then **ENTER**. Next, choose Raspberry Pi 3, and then get the default image when prompted. Make sure you have a microSD card ready to flash for when it asks for it to be inserted after the download. Follow along with the rest of the prompts to select the card and wait for it to be flashed.


C:\Users\darth\Downloads\android-things-setup-utility\android-things-setup-utility-windows.exe
What hardware are you using?
1 - Raspberry Pi 3
2 - NXP Pico i.MX7D
1
You chose Raspberry Pi 3.
Setting up required tools...
Fetching additional configuration...
Downloading platform tools...
4.74 MB/4.74 MB
Unzipping platform tools...
Finished setting up required tools.
Raspberry Pi 3
Do you want to use the default image or a custom image?
1 - Default image: Used for development purposes. No access to the Android Things Console features such as metrics, crash reports, and OTA updates.
2 - Custom image: Upload your custom image for full device development and management with all Android Things Console features.
1
Downloading Android Things image...
331 MB/331 MB
Unzipping image...
Downloading Etcher-cli, a tool to flash your SD card...
19.8 MB/19.8 MB
Unzipping Etcher-cli...
Plug the SD card into your computer. Press [Enter] when ready


C:\Users\darth\Downloads\android-things-setup-utility\android-things-setup-utility-windows.exe
Android Things Setup Utility (version 1.0.21)
This tool will help you install Android Things on your board and set up Wi-Fi.
What do you want to do?
1 - Install Android Things and optionally set up Wi-Fi
2 - Set up Wi-Fi on an existing Android Things device
2
What hardware are you using?
1 - Raspberry Pi 3
2 - NXP Pico i.MX7D
1
You chose Raspberry Pi 3.
Setting up required tools...
Fetching additional configuration...
Downloading platform tools...
4.74 MB/4.74 MB
Unzipping platform tools...
Finished setting up required tools.
Please plug your Raspberry Pi to your router with an Ethernet cable, then press [Enter].

Setting up wireless internet

STEP 04

Once flashing is done, you can take the microSD card out and put it into your Raspberry Pi. At the very least, you need to have the microSD card inserted and have access to a power supply. If you still have the setup utility open, you can then set up the wireless internet on your device by also plugging an Ethernet cable into the Pi. If you've exited the setup utility, just start it up again and choose the second option.

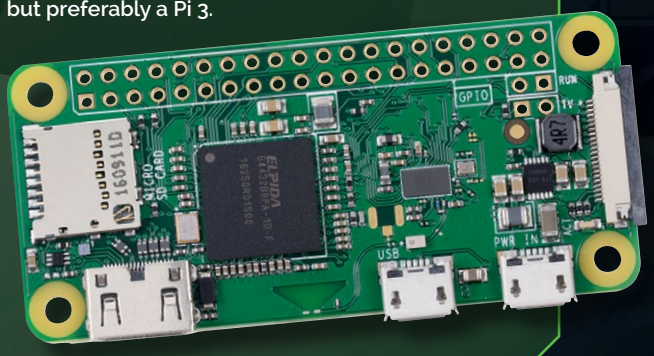
Alternate wireless setup

STEP 05

If you don't have access to wired internet, you can always plug the Pi into a monitor and use the included utilities to connect to your WiFi for further configuration. You can also do it via adb (Android Debug Bridge) if you want to get into the nitty-gritty of wireless setup – find out more info on that here: **magpi.cc/PHWvQR**.

Android Zero

Unfortunately, Android Things does not support the Pi Zero – it requires a bit more power than the board can currently provide, so you'll need at least a Raspberry Pi 2 to run Android Things on, but preferably a Pi 3.



USING ANDROID THINGS

Learning how to use the IoT-focused Android

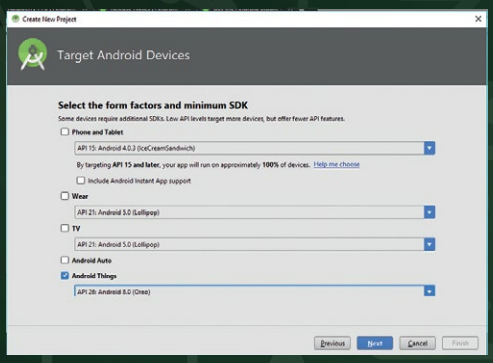
Programming Android Things is much like making an Android app. It can make use of Google services and is programmed in Java. This means it's a bit more advanced than using Python in Raspbian – even something as simple as lighting an LED takes a lot more time than using the Python GPIO Zero library to do so.

Software is built into Android Studio and then uploaded to your Raspberry Pi – this way you can quickly prototype and test your code before updating it and testing again. It also means you can have multiple projects on the go that use the same Pi – you just need to switch between them.

KNOW YOUR I/O

Setting up an Android Studio project

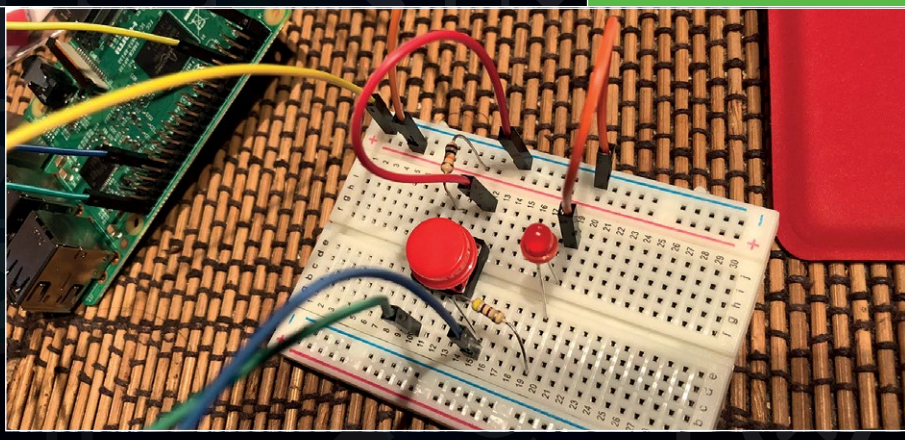
Grab the Android Studio software from **magpi.cc/DqPMKe**. Install it and then open it, and go to 'Start a new Android Studio Project'. Call it whatever you wish and then select Android Things as the form factor, with the latest SDK. You can then access your project through here, including any previous code you've created or edited.



GPIO SIGNAL

BCM2	I2C1 (SDA)
BCM3	I2C1 (SCL)
BCM7	SPI0 (SS1)
BCM8	SPI0 (SS0)
BCM9	SPI0 (MISO)
BCM10	SPI0 (MOSI)
BCM11	SPI0 (SCLK)
BCM13	PWM1
BCM14	UART0 (TXD) or MINIUART (TXD)
BCM15	UART0 (RXD) or MINIUART (RXD)
BCM18	I2S1 (BCLK) or PWM0
BCM19	I2S1 (LRCLK)
BCM20	I2S1 (SDIN)
BCM21	I2S1 (SDOUT)

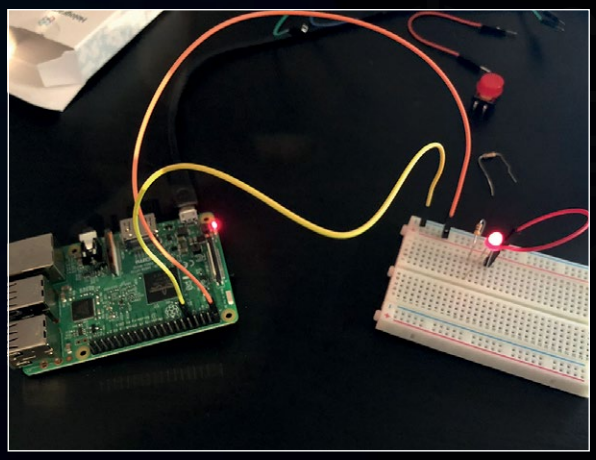
GREAT STARTER PROJECTS



LED blinker

magpi.cc/cKFQQw

This beginner's tutorial shows you how to blink an LED using Android Things. While it's much more complex than using GPIO Zero, you'll be using Android Things for more than just controlling electronics via the GPIO. It's well explained and gives you a nice basic tour of the Android Studio work area.



LED with a button

magpi.cc/FKSVLp

A follow-up tutorial to the LED blinker, this adds input command code to the already present output code for the LED. From here you can begin to see how the code reacts to events, and you can start building slightly more complex circuits as well.

Using Bluetooth

magpi.cc/QbDjXY

Jumping ahead, this tutorial teaches you how to make use of Bluetooth communication on your Raspberry Pi with Android Things, which is slightly more robust than the available Python libraries you might use in Raspbian. From here you can start truly developing an interesting project for yourself.

Bluetooth®

THINGS RESOURCES

Android Things documentation

The basic information for Android Things is a good place to look for all your Android Things needs – it also covers other devices, which may be of use to you if you're planning to branch out with what your Things app will work on.

magpi.cc/BraebS

Hackster.io Android Things projects

Many people use Hackster.io to post and document their builds, and there are plenty of projects to find on here for both beginners and advanced users alike. There's even a special Android Things tag for those kinds of projects, and you can further filter it by Raspberry Pi projects.

magpi.cc/gpmLrZ

Raspberry Pi Forums

Need some help with your Pi project? The Raspberry Pi forums are the best place to go to ask questions about what you're doing – you may even be lucky enough to find an existing thread that solves your specific issue!

raspberrypi.org/forums

Not sure what to make with Android Things? Turn the page for some inspiration...

AMAZING ANDROID THINGS PROJECTS

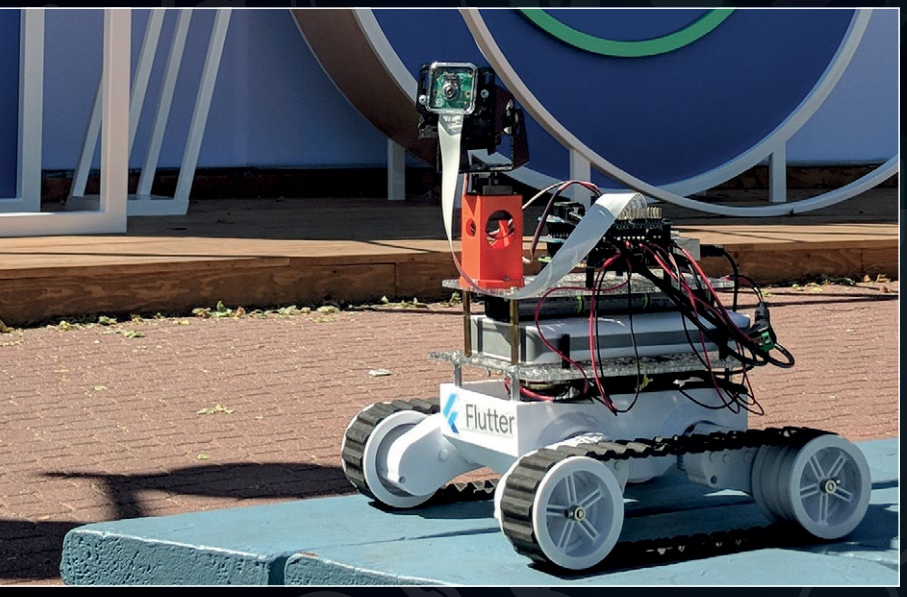
Not sure where to get started with Android Things? Here are some cool projects to help inspire you!

Sentinel

magpi.cc/YsxVco

Looking for a home security robot? Well, you might want to keep looking (and wait a few decades), but you can build this semi-autonomous Pi robot that's able to provide some level of security.

The Sentinel follows the maker's dog, as he wanted to see how it's getting on while he's not home. Sentinel was cheaper than IP cameras. Facial recognition has also been added, as well as text-to-speech so the robot can talk to you.



BrailleBox

magpi.cc/dxMNje

An ingenious project to help blind people stay informed, the BrailleBox at its simplest loads news stories from the internet and then translates them into Braille for the user to be able to read it.

The project has evolved beyond this initial concept, though, allowing the user to slow down or speed up news reading, and to press a button for the device to load the latest news story from News API (**newsapi.org**).

Lantern

magpi.cc/wZboeI

This recent project took the internet by storm, using an off-the-shelf IKEA lamp and turning it into a laser-projected smart interface. It's all high-tech and futuristic and runs simply on a Raspberry Pi 3 with Android Things.

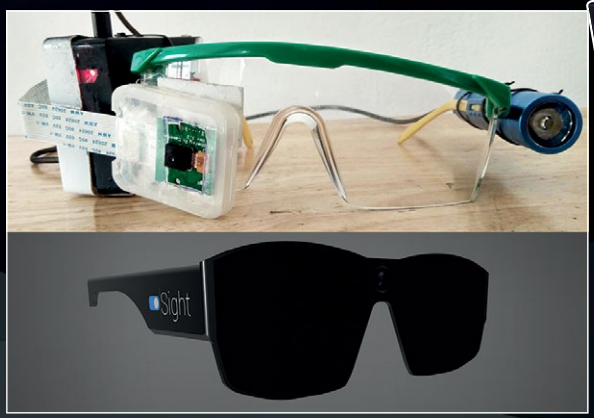
As well as smart interfaces, it can project what the makers call 'ambient data' onto a surface: say, a scrolling weather forecast onto a wall, or a 'now playing' info screen on the edge of a thick side-table.

Word Clock

magpi.cc/fbtunU

While analogue watches may be a great fashion statement, they're not quite as convenient as digital watches. Or this word clock, which writes out the time using letters cut out of a board.

You can control the colours of the LEDs through a dedicated companion app, or by connecting it to a Google Home assistant. Ideal for lowering the brightness while you watch a film or to set the nerdiest mood lighting imaginable.

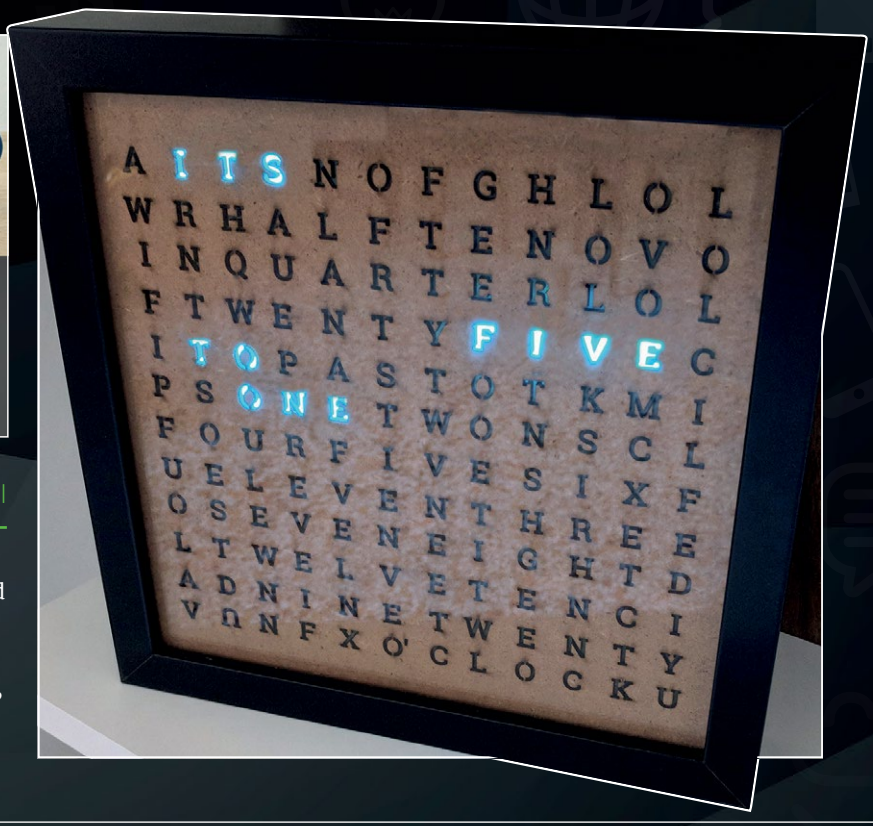


SIGHT

magpi.cc/buNCqI

Another project designed for people with vision impairments, these glasses look ahead of the wearer and tell them what's in front of them using Google services such as TensorFlow.

This could be modified to be used a bit like a futuristic, high-tech smart display, showing you what's ahead, or even to help create advanced automation in a robot.



SUBSCRIBE TODAY FROM JUST £5

SAVE UP TO 35%

£5 FREE! MODMYPI VOUCHER FOR ALL SUBSCRIBERS

Pricing

Rolling Subscription

- £5 a month
- Quick and easy to set up
- No long-term commitment

* Leave any time applies to Rolling Subscription only

Subscribe for a year:

£55 (UK)

£80 (EU)

£90 (USA)

£95 (Rest of World)

Subscription benefits:

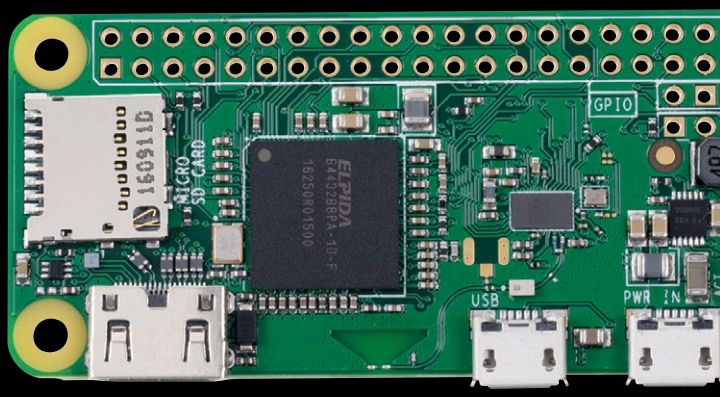
- **FREE!** Delivery to your door
- **EXCLUSIVE!** Raspberry Pi offers and discounts
- **NO OBLIGATION!** Leave any time*

magpi.cc/subscribe

JOIN FOR 12 MONTHS AND GET A

PI ZERO W STARTER KIT

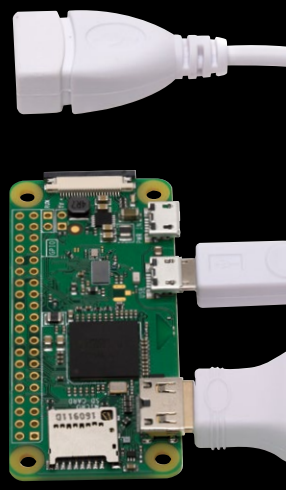
WITH YOUR SUBSCRIPTION



WORTH £20

Subscribe in print for 12 months today and you'll receive:

- Pi Zero W
- Pi Zero W case with three covers
- USB and HDMI converter cables
- Camera Module connector



SUBSCRIBE ON APP STORES

Available on the App Store

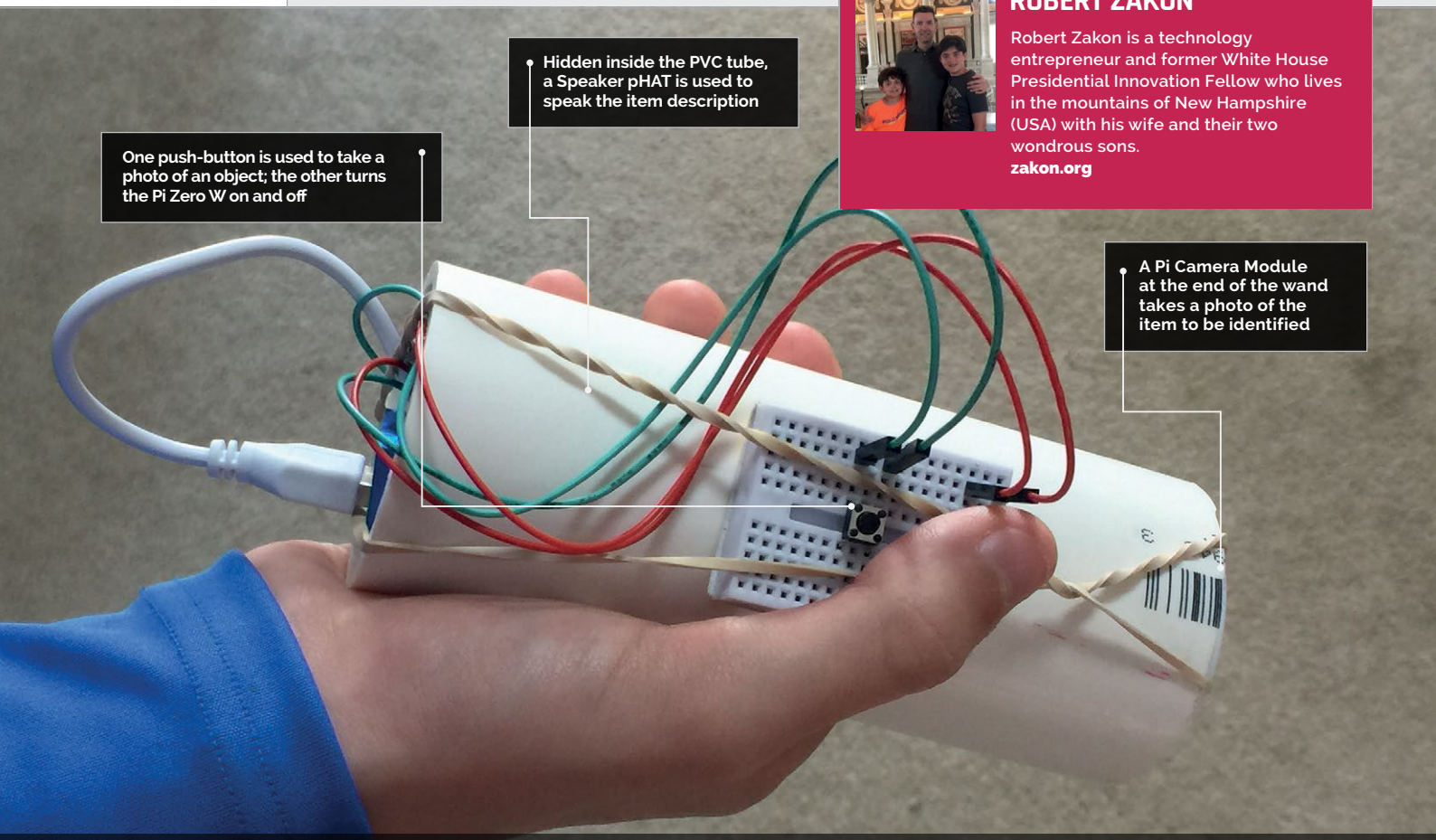
GET IT ON Google Play

FROM £2.29

BUY IN PRINT WORLDWIDE! magpi.cc/store
The MagPi
RUN ANDROID ON RASPBERRY P
MAKE YOUR OWN OSCILLOSCOPE
MAGIC: THE GATHERING CARD COUNTER
BUILD A WILDLIFE CAMERA
ADD AI TO YOUR PROJECTS WITH TENSORFLOW
HOME AUTOMATION WITH RASPBERRY PI
WIN!
UPCYCLE OLD TECH

ROBERT ZAKON

Robert Zakon is a technology entrepreneur and former White House Presidential Innovation Fellow who lives in the mountains of New Hampshire (USA) with his wife and their two wondrous sons.
zakon.org


Hidden inside the PVC tube, a Speaker pHAT is used to speak the item description
One push-button is used to take a photo of an object; the other turns the Pi Zero W on and off
A Pi Camera Module at the end of the wand takes a photo of the item to be identified

SEEING WAND

Quick Facts

- Build details are at **magpi.cc/FheQWt**
- The wand is programmed using Python
- Code is on GitHub: **magpi.cc/EQurCy**
- A Microsoft cloud API is used for ID
- Mistaken identifications can be amusing

Point this magic wand at an item and it'll speak its name. **Phil King** lifts the curtain to see how it's done

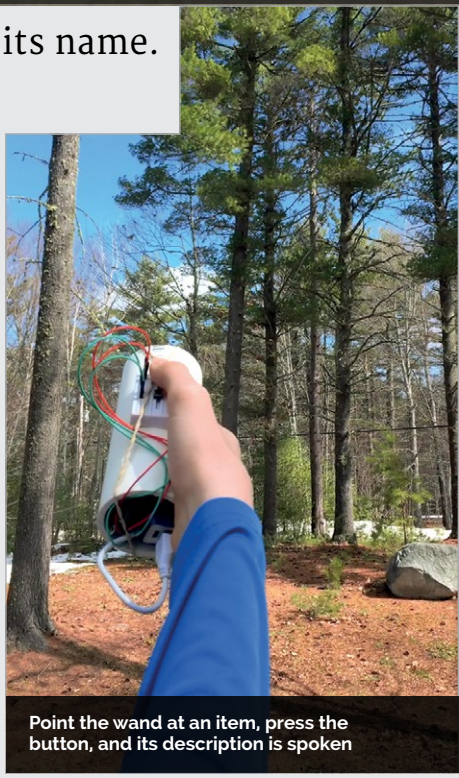
Inspired by a blind cousin who would 'look' around his environment by way of touch, Robert Zakon has built a Seeing Wand that can speak the name of whatever it's pointed at. Housed in a makeshift PVC tube, a Pi Zero is connected to a Camera Module that takes a photo when a push-button is pressed. The image is sent to Microsoft's Cognitive Services Computer Vision API to get a description, which is then spoken – using the open-source eSpeak speech synthesizer – through a Speaker pHAT.

"I was looking for a way to teach my kids about innovation through integration and had been wanting to test out both the Pi and emerging cognitive computing services," explains Robert. "They were a bit sceptical at first, but warmed up to it and thought the end result was pretty awesome (their words). My eldest helped with assembly, and both aided in testing."

First taste of Pi

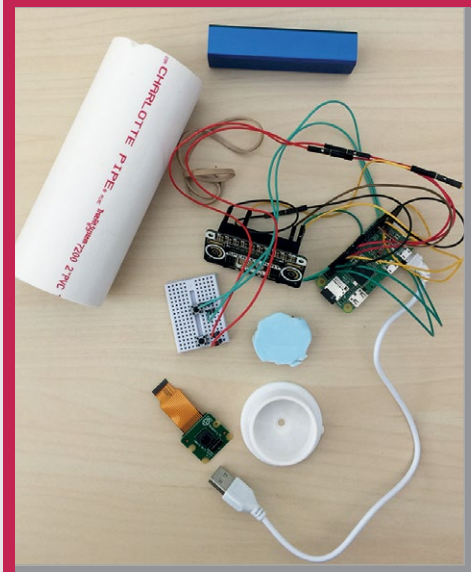
Robert's debut Raspberry Pi project, it came together over the course of a few weekends.

Asked why he chose Microsoft Cognitive Services over other image-recognition APIs, Robert responds: "Microsoft did a nice job with the API and it was fairly straightforward to integrate with. There was no particular reason for choosing it other than it appeared



Point the wand at an item, press the button, and its description is spoken

MAKE A MAGIC WAND

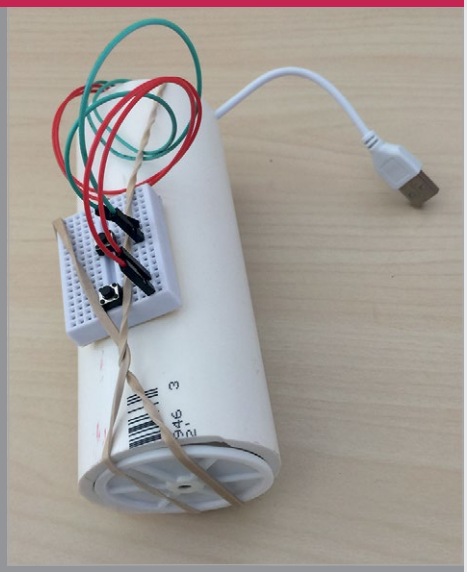


>STEP-01
Wiring the electronics

Components include a Pi Zero W, Camera Module, and Speaker pHAT. Wiring is currently via a mini breadboard. The device is powered by a 2200 mAh power cube.

>STEP-02
PVC housing

The electronics are crammed into a PVC tube. The camera fits into a closet-rod-supporting end cap and is held in place by rigid insulation, with its lens up against the cap's screw hole.



>STEP-03
Two buttons

The breadboard holds two push-buttons: one to take a photo of the item you want to identify, and the other – wired to the GPIO 03 and GND pins – to turn the Pi Zero W on and off.

to be robust enough and free to use for our project."

The results surprised him in terms of accuracy and level of detail: "People, pets, and large objects seem to be the sweet spot."

As per its original inspiration, however, the Seeing Wand could be of serious use to partially sighted people. "Although there are smartphone apps that do the same thing, this could be a less

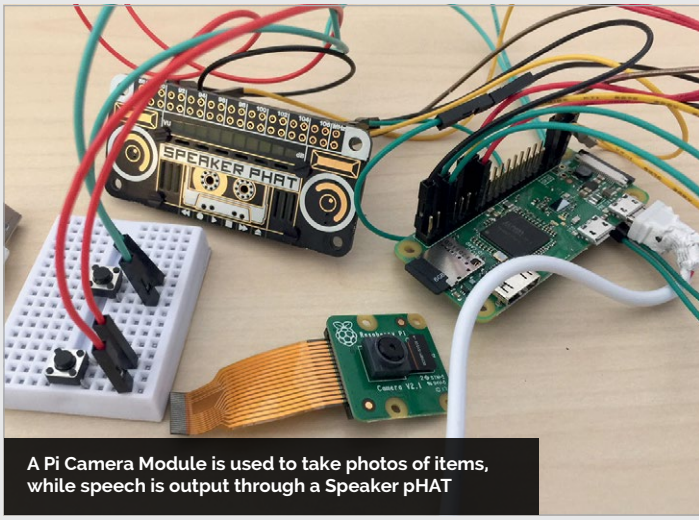
> "Even when the wand gets it wrong, the results can be amusing"

Even when the wand gets it wrong, the results can be amusing. "My kids had a lot of fun whenever something was misidentified, such as pointing at a toy robot on a table and having it identified as 'a small child on a chair'. Another example was pointing at our garage with a sloping roof and being informed there was 'a skateboarder coming down a hill' – still not sure what it thought the skateboarder was. My favourite, though, had to be when we pointed it at clouds and heard what sounded like 'Superman flying across a blue sky'."

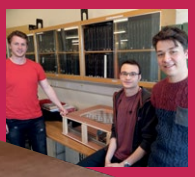
expensive and more human-friendly device."

Fine-tuning

Robert admits that the prototype wand is a little rough around the edges. "We have talked about making improvements both to the hardware and software. On the hardware side, we would solder all wires and buttons, and use a smaller battery in order to make it truly palm-sized and thinner so it could fit as the holding end of a white (blind) cane. For the software, we'd like to integrate the text recognition and possibly language translation services so signs and printed material could be read, and the face recognition service so people could be identified. Also, as the cognitive services are not yet perfect, it would be interesting to 'poll' multiple services and determine which identification is best through our own cognitive meta-service."



A Pi Camera Module is used to take photos of items, while speech is output through a Speaker pHAT



TIM NESS, ALEX ANGELOV, AND ALEX SMITH

Having just completed their final exams, the three students have the world at their feet as they seek graduate opportunities in industry.
magpi.cc/LxbagV

A mechanical arm below the board moves the pieces. By shifting them to the corner of a square and then only moving along the square edges, it avoids collisions with other pieces

Each chess piece has a button magnet on the bottom. The magnetic field needed to be strong enough to hold the piece while not attracting surrounding ones

A Raspberry Pi 3 runs the Stockfish chess engine. This allows the computer to decide on its moves, with software also controlling the robotic arm under the board

GHOST CHESS

Quick Facts

- Each chess piece is 3D printed
- Forty 3 mm button magnets are used
- The arm's electromagnet has a 4.5G pull force
- It's controlled using GPIO on the Pi
- Chess engine Stockfish is open-source

With this game of chess by Tim Ness, Alex Angelov, and Alex Smith, the pieces appear to move by themselves. **David Crookes** checks it out

On 10 February 1996, a chess-playing computer called Deep Blue sent shock waves around the globe by beating world champion Garry Kasparov in the first of their six games. But the IBM machine did so without moving any physical pieces itself – a human noted the computer's move and performed it manually on the board. Had Ghost Chess been around back then, however, such intervention would have been redundant.

Designed by final-year MEng students Tim Ness, Alex Angelov, and Alex Smith from the University of Glasgow, Ghost Chess makes use of a robotic arm connected to a Raspberry Pi running the world champion chess program, Stockfish (**stockfishchess.org**). It focuses attention on a physical board, using motors and an electromagnet to pick up, move, and place 3D-printed chess pieces within the squares depending on the moves dictated by both human and computer.

As such, it's a mini-marvel – a prime example of a real-time embedded system. "We wanted to create something that was fun, memorable, and slightly more challenging – an 'automatic' 3D chess game from scratch," says Tim. What's more, they wanted to make it as unobtrusive as possible. "We kept the robotics underneath the board because it was an easier way of designing the system," Tim adds. "It keeps all of the electronics and the moving mechanism out of the way."

Multilayered project

Indeed, the project is broken down into five layers: the chess pieces, the board, a matrix of sensors, the mechanical arm, and, at the bottom, the Raspberry Pi 3. Of

those, the arm was the trickiest part to develop. "Creating a design without having access to expensive ball races/bearings was a real challenge," Tim recalls.

T-slot bars bolted to a plywood base act as runners for two shuttles and these are hooked up to timing belts mounted through a couple of pulleys to a stepper motor, allowing the arm to move left, right, up, and down. Meanwhile, a matrix of 64 latching Hall effect sensors (one for each square and capable of varying the output voltage in response to a magnetic field) lets the setup detect which spaces on the board are occupied.

> "There was a lot of difficulty integrating all of the different parts of the system"

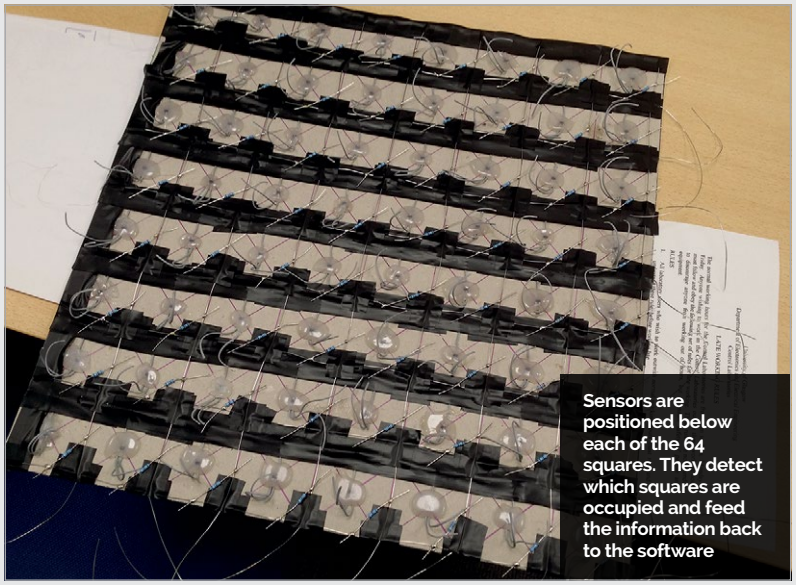
From that point on, it's up to the software running on the Pi to work its magic. For this, the students coded their own program to work with the Stockfish API. It not only allows humans to enter their moves through a graphical user interface or via a command line, it also empowers the computer to decide where it needs to go.

"The RPi is a versatile platform for development which provides competitive input and output capabilities, compared to other devices," says Tim. "We used Raspbian as the platform for development, which allowed us to create a GUI fairly easily and helped a lot with the development and debugging process."

Moving pieces

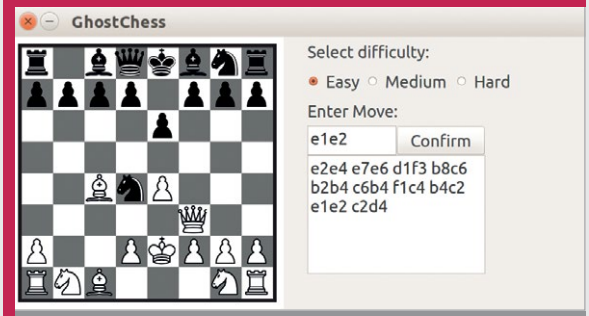
Once a move is decided, the software instructs the arm to make its way to a piece. It then turns on the electromagnet positioned on top of the arm, which attracts a button magnet fitted to the bottom of each piece. At this stage, the robotic arm can start to perform the move. Seeing physical pieces move across the board is far more satisfying than simply playing on a computer.

"There was a lot of difficulty integrating all of the different parts of the system and calibrating them in order to work as expected," says Tim. "We think the trickiest part was trying to work on a tight budget and design solutions for different parts of the system, without breaking the bank, and we hope the idea will be passed on and upgraded during the coming years."



Sensors are positioned below each of the 64 squares. They detect which squares are occupied and feed the information back to the software

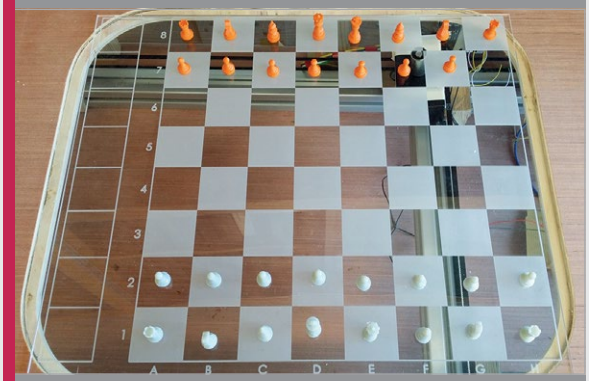
TIME TO MAKE A MOVE



>STEP-01

Enter the move

Human players can make a move using Ghost Chess's graphical user interface via an attached screen. The control software uses a fixed reference point as the origin of the board: the a1 square.



>STEP-02

Calculating positions

The co-ordinates of the piece to be moved are given and the distance to travel to reach that square is calculated. The arm is moved with stepper motors, so distances are calculated as steps.

>STEP-03

Performing the action

The arm moves into position and the electromagnet is activated, attracting the piece. The distance to the new square is calculated. Once the move is made, the electromagnet deactivates and the arm returns to a1.

MICHAEL PORTERA

Michael is a cybersecurity manager who loves to create innovative projects with Raspberry Pi boards and build skills for Amazon Alexa.
magpi.cc/fAlqgq

TRADING CARD SCANNER/ORGANISER

For his next trick, **Michael Portera** created a system out of LEGO to recognise, organise, and value his vast collection of Magic: The Gathering trading cards. **David Crookes** investigates

Quick Facts

- The build and code took Michael ten hours
- It uses LEGO bricks – and plastic tyres
- The various components, Pi included, cost $115
- 20–25 cards can be recorded per minute
- There are 20 000 official MTG cards

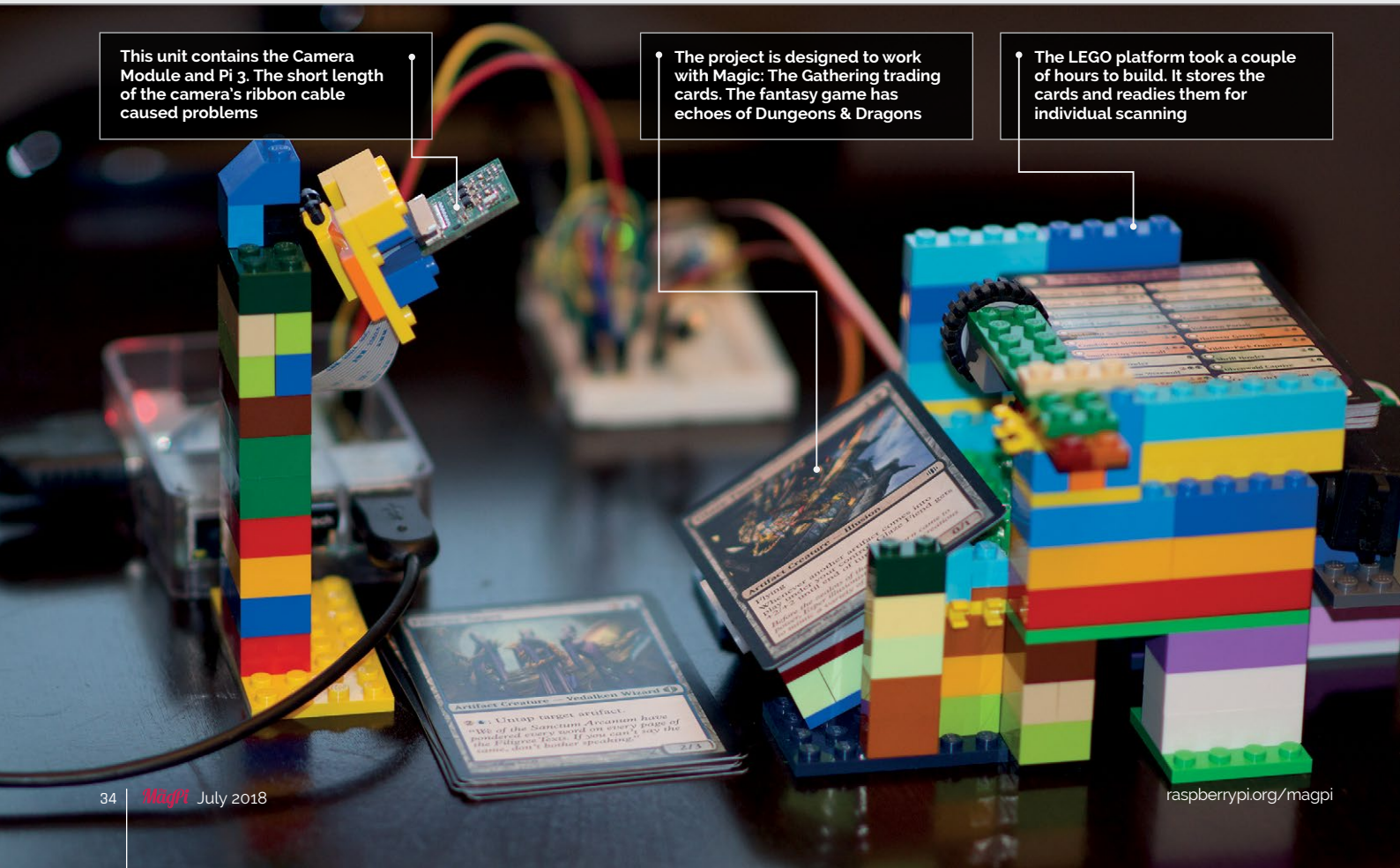
When Michael Portera came across a few boxes packed with trading cards – "everything from football, baseball, and basketball to Magic: The Gathering (MTG)" – one of his first thoughts was how much the collection might be worth. As someone who had dabbled with MTG by buying booster packs and playing the occasional game with friends, he knew there was a large secondary market and that he had cards worth investigating.

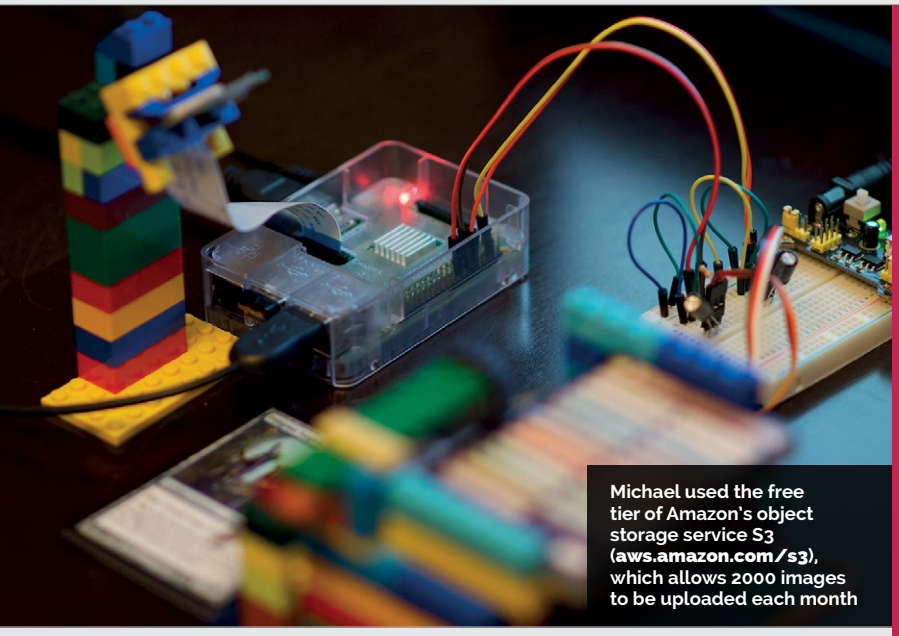
"But looking up the cards manually would have taken a while, even using online scanners or apps," he muses. "So I wanted to develop an automated process using a card feeder and a scanner for image processing."

Motivated by the potential time saved, he quickly got to work. The spell-binding result was a scanner made of LEGO, servo motors, a camera, and a Raspberry Pi 3.

Shuffling cards

Right from the start, Michael envisaged a simple system that would spin and push preloaded cards forward to be photographed and uploaded for digital storage and processing. The idea was to make light work of organising and valuing even the largest of collections.


This unit contains the Camera Module and Pi 3. The short length of the camera's ribbon cable caused problems
The project is designed to work with Magic: The Gathering trading cards. The fantasy game has echoes of Dungeons & Dragons
The LEGO platform took a couple of hours to build. It stores the cards and readies them for individual scanning



Michael used the free tier of Amazon's object storage service S3 (**aws.amazon.com/s3**), which allows 2000 images to be uploaded each month

Building it, however, was largely a process of trial and error.

"I had a cheap card shuffler lying around," recalls Michael. "After taking it apart, I found the simple cog system powered by a DC motor that moves the hammer and pushes one card forward into a slot, creating a new stack of cards. This would be the inspiration for the servo and wheels needed to accomplish the automation."

Box of bricks

Michael turned to LEGO for the main structure due to its versatility. It allowed him to easily build, modify, and tear down his project. He placed servos in the back of the build and had them spin continuously. Carefully positioned LEGO tyres then move forward in a cog-like setup to get the cards into position.

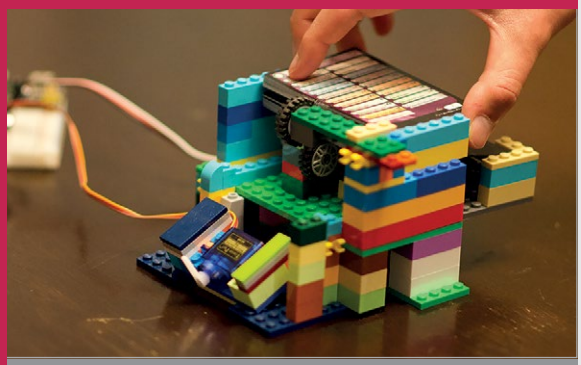
Once the device was built and he was happy, Michael could then start coding, which he says was the easiest part of the build. He used Python 2.7 to program a script to power both the servos and take a picture via a Raspberry Pi Camera Module. He then wrote another script to send the images to Amazon's cloud computing web service S3, for storage and to tackle image processing.

"I originally tried Tesseract and OpenCV for optical character recognition, but I spent a lot of time trying to get so many variables perfect and I couldn't get consistency," he explains. To fix this, he turned to Amazon's deep learning-based image analyser, Rekognition (**magpi.cc/sfLJLE**), which extracts text and indexes a collection. "It works well. I did not have to worry about getting the angle just right, making sure the lighting was perfect, or performing any machine learning – it worked despite all of those factors."

> "The idea was to make light work of organising and valuing even the largest of collections"

Apart from issues deciphering some fonts, it performed well: 619 of the 920 cards scanned perfectly and he was able to feed the data through TCGPlayer.com's price data API to determine the value of each card and, therefore, his MTG collection. "I had about $275 worth of commons, uncommons, and rares," he says, pleased as punch. "And through trial and error, I also learned a lot along the way."

TRADING TIME FOR CREATIVITY



>STEP-01

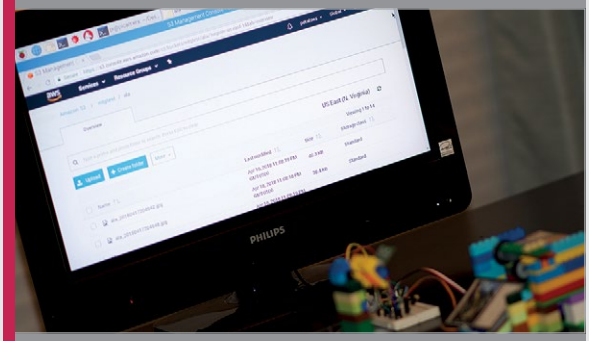
Insert the cards

Load your cards into the LEGO device and run a script on the Raspberry Pi to move the rear servos and allow a single card to get into position. A front wheel stops the other cards slipping.

>STEP-02

Snap an image

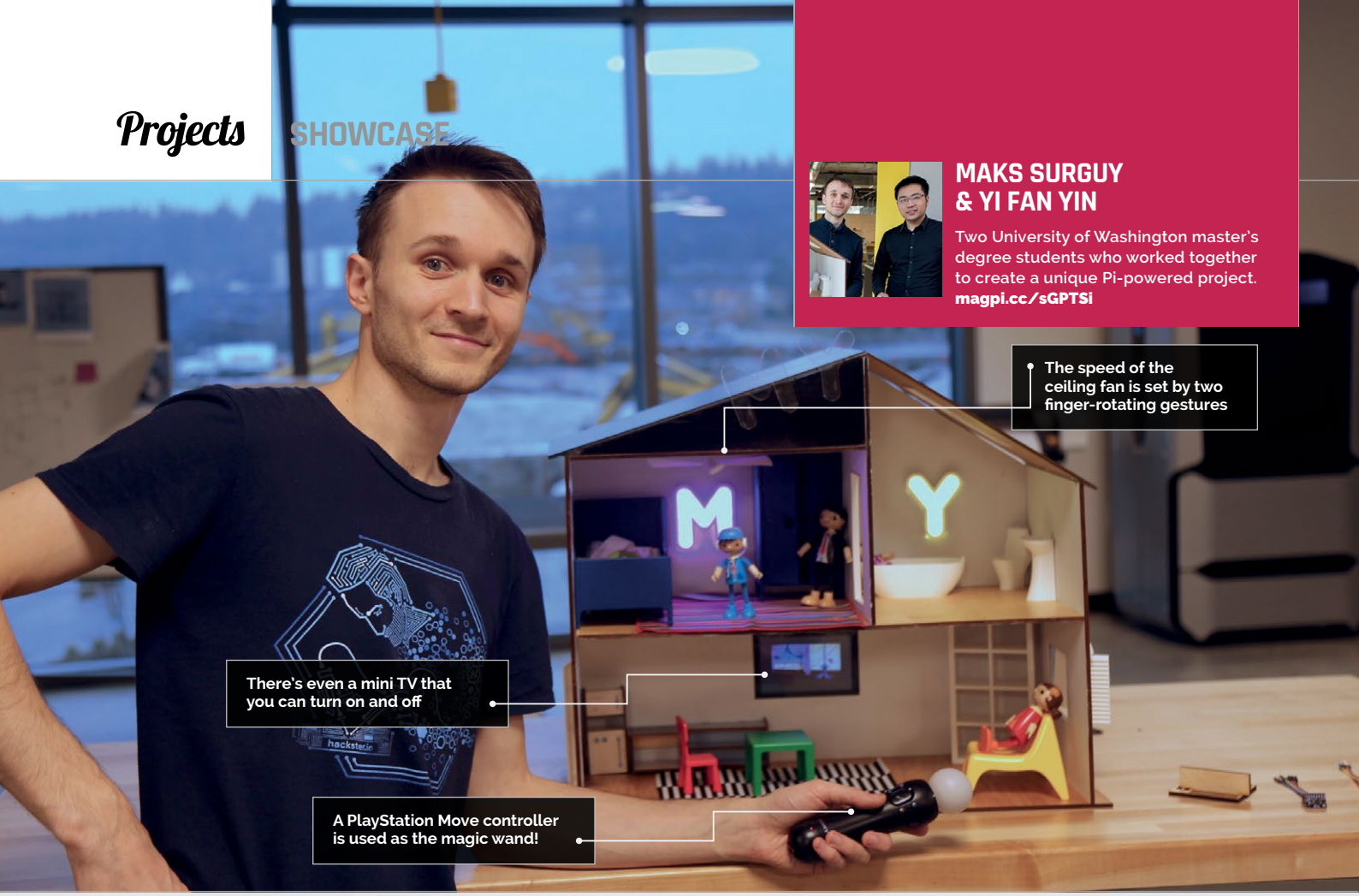
Once the card is in the scanning area, a Raspberry Pi Camera Module – carefully angled and sitting on a stack of bricks mere inches away – captures the top part of the card. Stop the script.



>STEP-03

Performing the action

Drag and drop the scanned card files into Amazon S3's online interface via the Pi. Run another script to get Rekognition to analyse the images and look up the market price.


There's even a mini TV that you can turn on and off
A PlayStation Move controller is used as the magic wand!
The speed of the ceiling fan is set by two finger-rotating gestures

MAKS SURGUY & YI FAN YIN

Two University of Washington master's degree students who worked together to create a unique Pi-powered project.
magpi.cc/sGPTSi

Quick Facts

- The doll's house took ten weeks to design and prototype
- A Pi 3 reads information from a motion sensor
- 'MYHouse' is short for 'Maks and Yi Fan's house'
- All the code can be found at **magpi.cc/GnJNjN**
- Machine learning is used to 'train' the gestures

PROJECT MYHOUSE

This smart doll's house features gesture recognition. **Nicola King** takes a look inside...

When a master's degree course at the University of Washington required the use of sensors and machine learning in the same project, two students – Maks Surguy and Yi Fan Yin – conceived the idea of an interactive doll's house. Inside this cool crib, various features – including lighting and shutters – can all be turned on and off by the simple wave of a 'wand' (a PlayStation Move controller), with the help of some clever coding and a Raspberry Pi 3. You can see a demonstration video at **youtu.be/6EiTWZfPm3k**.

"I thought a smart doll's house would be a great tool to demonstrate technical innovations to people in an approachable way," says Maks, who worked with Yi Fan over a ten-week period, designing and constructing the clever little doll's domicile.

After consulting Maks's architect wife about the physical structure, the pair drew the plans in 3D modelling software, then fitted together cardboard pieces for a prototype. Once happy with the design, they laser-cut the pieces out of plywood, made use of snap-fit to join them, then painted them in different colours.

According to Maks, building a doll's house is akin to building a real house. "Lots of decisions needed to be made about dimensions, colours, structure, function, and interactions between all elements of the dollhouse. We ended up simplifying a lot of the elements through iterative process after realising that what we envisioned is actually a lot harder than it seems. Thankfully we had 24/7 access to a makerspace here in school and were able to reach decisions through prototyping every aspect of the construction."

Grand gestures

A key characteristic of this smart doll's house is its ability to

recognise gestures and respond accordingly. Having done a great deal of research into gesture recognition, "trial and error went into choosing what gestures perform best across individuals while remaining intuitive to most people," says Maks. "We read a lot of research papers on gesture recognition and then came up with our own gestures that worked with over 90 percent accuracy."

In total, seven gestures – pre-trained using machine learning – are stored in the system, and the Raspberry Pi reads the information from the PlayStation Move and then determines if the gesture is similar to one of the stored ones.

The machine learning aspect of the project also presented certain challenges, as Maks recalls: "We ran into trouble selecting the most intuitive gestures and had to do quite a bit of trial and error to refine the experience. It takes about 20 samples per gesture to train the neural network, which is doable in a matter of a few minutes."

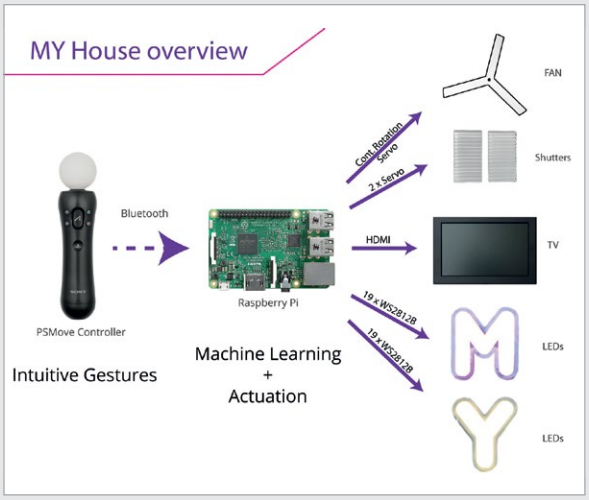
Possibilities of Pi

Maks was keen to use a Raspberry Pi in this project as he is enthusiastic about the possibilities it presents: "I am interested in pushing the Pi to its limits." He also has plans for the future, currently working with the Processing Foundation as a part of Google Summer of Code initiative to reduce barriers in using the Processing language on the Raspberry Pi. "My plan is to create a comprehensive resource that teaches people of all ages how to use the Raspberry Pi and Processing together, taking advantage of all connectivity and interactivity available on the Pi."

> "I am interested in pushing the Pi to its limits"

As Maks explains, if the gesture is recognised, "various functional items in the dollhouse can be activated or deactivated using these pre-trained gestures: TV, lights, fan, and shutters."

If the idea of an interactive doll's house appeals, the open-source nature of the code that Maks has created means that this is a build that anyone can attempt, as long as they possess some coding skills and the necessary components. "We haven't released the building plans for the dollhouse yet but if somebody's interested, I can share those as well," says Maks.



Above A Raspberry Pi 3 reads the gestures and controls all the electronic devices in the house

BUILDING A CLEVER DOLL'S HOUSE

>STEP-01
Plywood structure

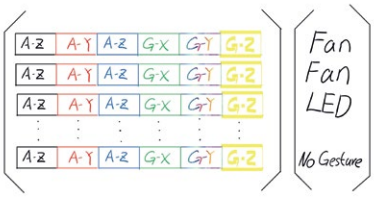
After constructing a cardboard prototype, the doll's house was built from laser-cut plywood pieces which snap together.

>STEP-02
NeoPixel letters

The laser-cut M and Y letters are each fitted with a strip of NeoPixel LEDs. Servos are used to rotate the ceiling fan and open the shutters.

>STEP-03
Gesture training

Maks and Yi Fan researched the most intuitive gestures to use. Each gesture was trained using a neural network, which involved taking around 20 samples.

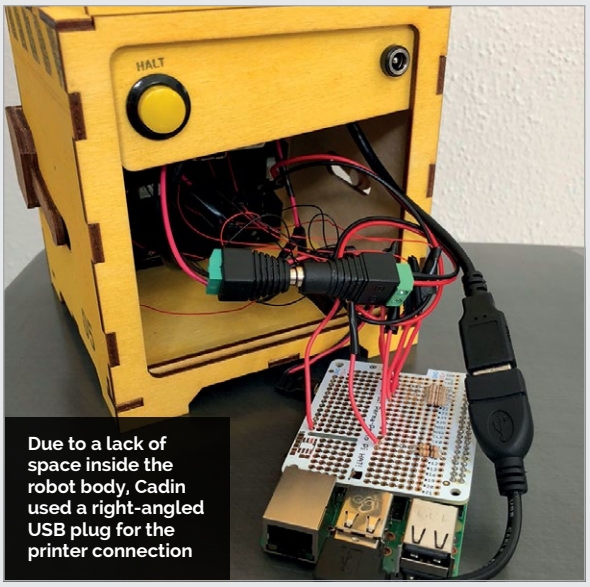


CADIN BATRACK

A Seattle-based interaction designer and developer, Cadin enjoys working on projects that use both his design expertise and programming skills.
magpi.cc/DzLYlL

VOMIT COMIC ROBOT

A tiny robot designed to make you smile? **Nicola King** sees the funny side of robotics



Due to a lack of space inside the robot body, Cadin used a right-angled USB plug for the printer connection

Quick Facts

- It took three months to design and build the robot
- Both Python and Processing are used
- View a video of it in action at **magpi.cc/JXhmpT**
- See Cadin's latest comics on Instagram **@cadinb**
- He'd like to make a themed comic version of the robot

Cadin Batrack enjoys creating his own small comics, and likes to publish a new one every day on his Instagram account (**@cadinb**). Having decided to create a program to randomly generate high-resolution comics, he was inspired to take things a step further.

"Shortly after starting that project, I was invited to participate in a local comic show," Cadin tells us. "I wanted a way to show off the random comic software at my booth, to let people generate and print a custom comic to take with them. The obvious solution was to build a little yellow robot that vomits comics out of his mouth."

The case is constructed from laser-cut plywood
Press this button to generate a comic
A thermal receipt printer produces the comic

Out of the mouths of robots

Having created the random comic software using the Processing language, Cadin chose to use a Raspberry Pi 3 to run this and a Python script. The latter controls the robot's LEDs, handles push-button input, and sends the comic images to a mini thermal printer connected to the Pi via USB. Cadin made a laser-cut plywood casing to hold all of the components, and the amusing little 'Vomit Comic Robot' was born.

When the robot's button is pressed, the Python script sends a message to the Processing sketch. "The sketch generates a random comic layout, and chooses some of my drawings to populate the frames," explains Cadin. "It chooses at random from about 100 drawings. Each image has some associated data that informs how it should be composed within a comic frame. The final composite image is saved to the Pi's SD card."

The Python script then loads the image and sends it to the printer. "With each comic I also print the date, the comic number, and the name of the event."

> "I would like to make a version that lets you choose a theme for your comic, like 'food' or 'animals'"

The robot that gags... gags, geddit?

Cadin hopes to tweak the robot in the future: "I would like to make a version that lets you choose a theme for your comic, like 'food' or 'animals' or 'nature'. I imagine a big dial on the robot that lets you select the theme and then you get a comic with images from that theme when you press the button."

According to Cadin, the project could easily be replicated using his comic-generating code (**magpi.cc/tOYOcL**), and makers can add their own drawings. "If you wanted to print something else (not random comics) it would be even easier, because you could leave out the Processing sketch entirely."

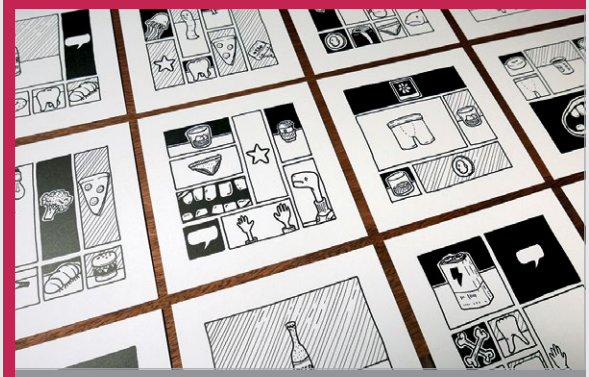
Cadin has received a lot of positive feedback on his humorous hack, but says he isn't sure that people really understand what is happening inside the robot: "It would be nice if they could somehow see the comics being generated to understand that it's happening dynamically, and not just printing out a pre-made comic."

Some people have suggested adding more narrative to the comics, but Cadin isn't sure: "I like that they come out as nonsense... He's not a very smart robot!"

Vomit Comic Robot may not be smart, but this is one quipster guaranteed never to get stage-fright.

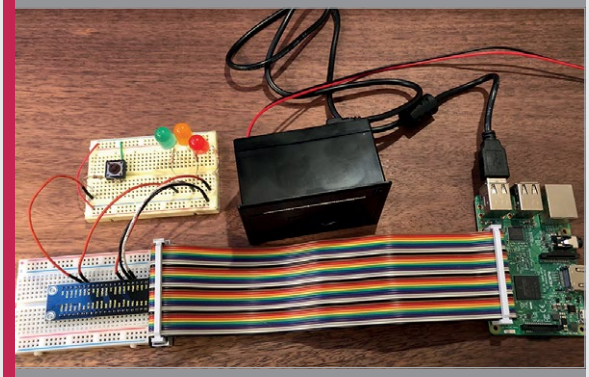
Each printout features details including the event name and date

MAKING A VOMIT COMIC ROBOT



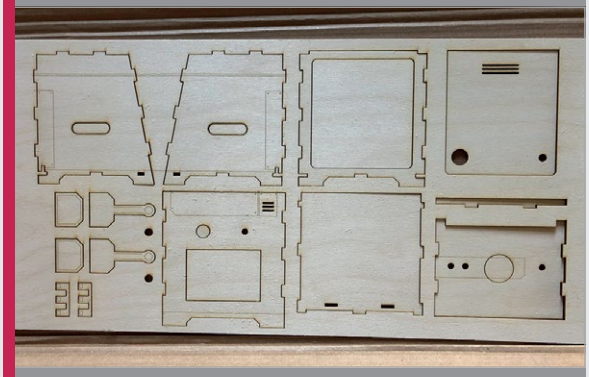
>STEP-01
Creating comics

Cadin first created the random comic generation software using the Processing language. You can find his code on GitHub at **magpi.cc/tOYOcL**.



>STEP-02
Connecting the components

All the software runs on a Raspberry Pi 3. Its GPIO pins are connected to a push-button, status LEDs, and the thermal printer.



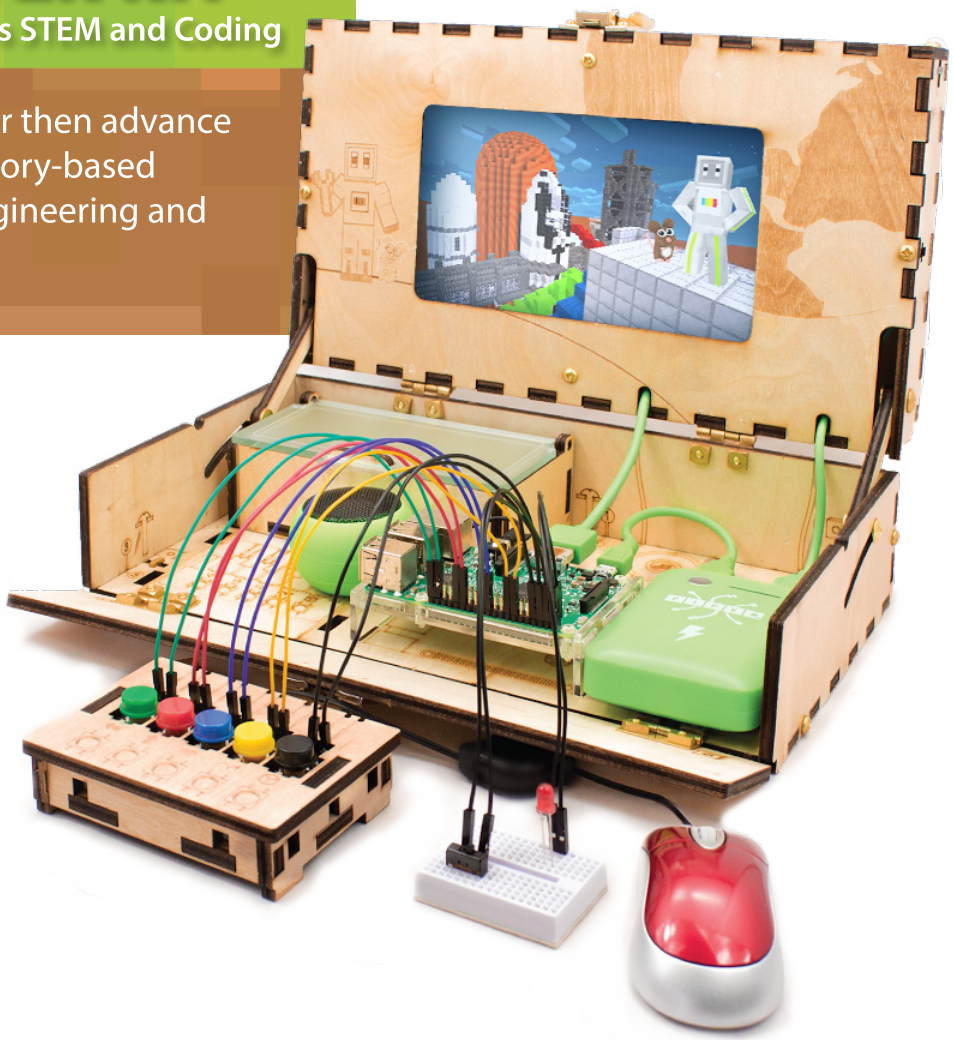
>STEP-03
The final build

Laser-cut plywood was used to construct the robot's body. Cadin admits he made it slightly too small, so had to make some adjustments inside.

PIPER COMPUTER KIT

Educational Computer that teaches STEM and Coding

Kids build their first real computer then advance through Piper's award-winning story-based curriculum and learn physical engineering and electronics in the process.



Special $10 MagPi coupon:

MagPiSummer

only at BuildPiper.com

"It's a really fun educational computer kit that should really impress those who love Minecraft and building stuff" — The MagPi

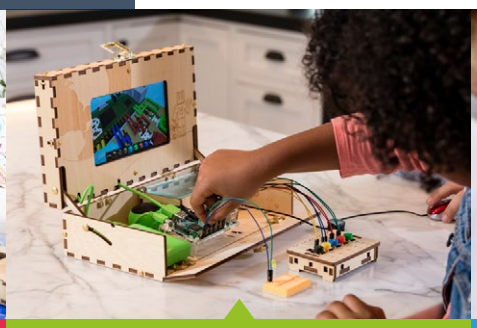
A STEM SOLUTION FOR SCHOOLS

Teachers all over the world use Piper to inspire kids to program, design, and engineer.

Want to learn how to bring Piper to your school?
buildpiper.com/EDU

THE PIPER EXPERIENCE

BUILD FROM SCRATCH



SOFTWARE-BASED CURRICULUM

CREATE YOUR ELECTRONICS

Supported by Top University Funds
Stanford-StartX Fund
AEF of Princeton University

Available at:

PIPER BuildPiper.com · amazon · Toys"R"Us · BARNES&NOBLE

ACCESS YOUR RASPBIAN DESKTOP REMOTELY USING VNC

Use your Raspberry Pi from any other machine on your network

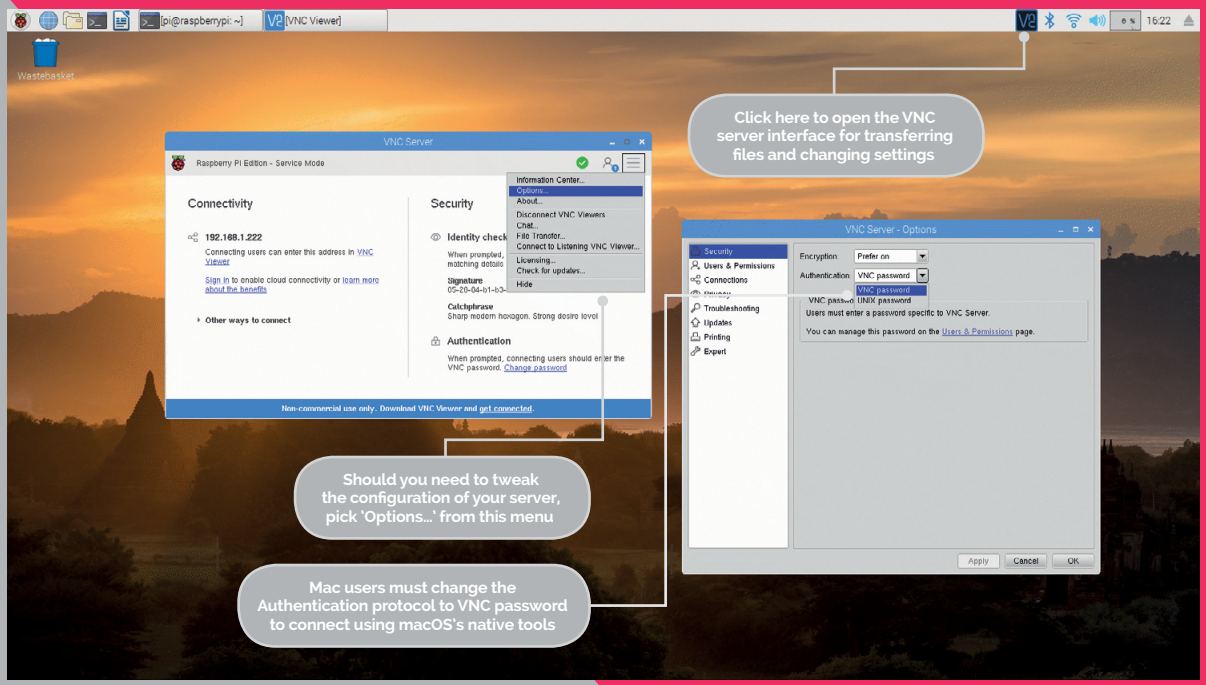
You'll Need

- Raspberry Pi running Raspbian
- Network connection
- Another computer on your network

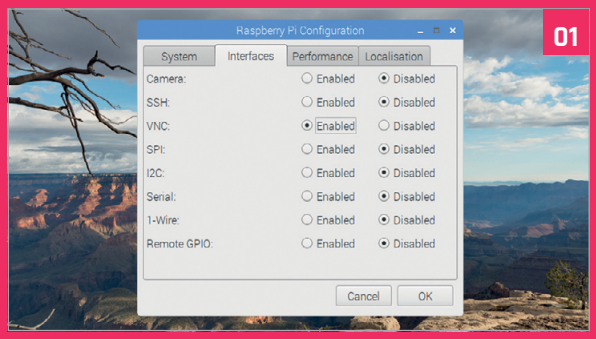
There will be times when you can't or don't want to switch to your Raspberry Pi. Perhaps you're using another computer, or your Raspberry Pi is out of reach, behind your TV or a nest of cables. Fortunately, with VNC (Virtual Network Computing), free for non-commercial use and built into the Raspbian operating system, you can access the Pi remotely from any other computer, tablet, or smartphone on your network.

In this walkthrough, we'll be using VNC Viewer to connect on the fly from one Raspberry Pi to another. If it's something you'll do frequently, set up a free RealVNC account at **magpi.cc/ctHuUR** and use this to sign in on up to five devices through the VNC Viewer. Doing so will save you entering their IP addresses in Step 3.

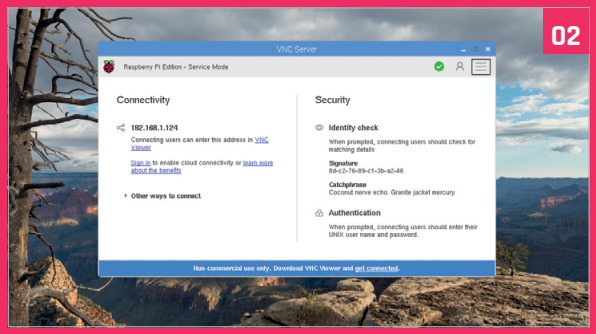
RealVNC produces clients for Linux (including Raspberry Pi), Windows, and macOS. Mac users can connect using macOS's built-in tools. To do this, switch to the Finder by clicking the first icon on the Dock, then press **COMMAND+K** and enter **vnc://0.0.0.0/** – replacing 0.0.0.0 with your Raspberry Pi's IP address (see Step 2). You'll also need to change the authentication method on your Raspberry Pi by clicking the VNC button on the Raspbian toolbar, selecting 'Options…' from the menu, and setting Authentication to VNC password, as we've done in the screen grab below.


Click here to open the VNC server interface for transferring files and changing settings
Should you need to tweak the configuration of your server, pick 'Options…' from this menu
Mac users must change the Authentication protocol to VNC password to connect using macOS's native tools

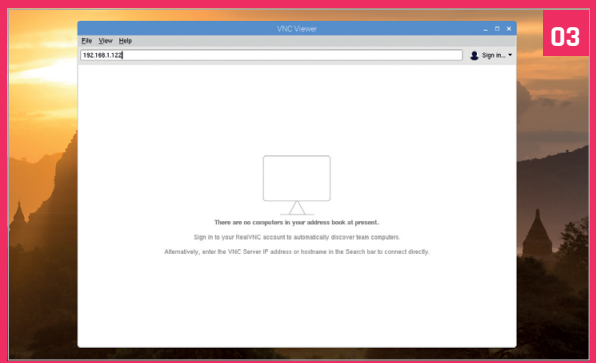
HOW TO: SET UP VNC IN RASPBIAN



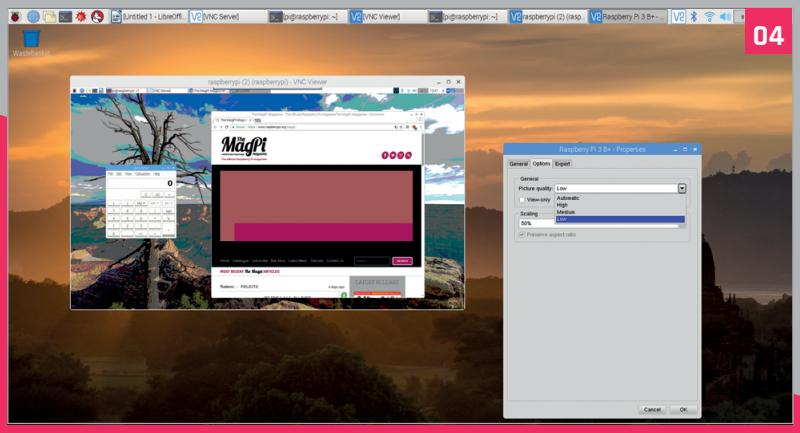
>STEP-01
Enable VNC
Click the Raspberry Pi icon at the top-left of the screen and select Preferences > Raspberry Pi Configuration. Click the Interfaces tab, followed by the Enabled radio button beside VNC.



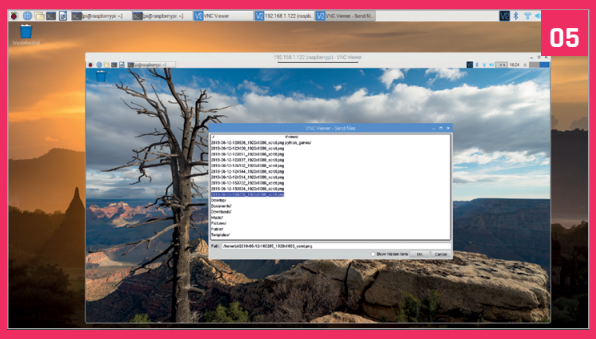
>STEP-02
Check your credentials
Click the VNC button that appears at the end of the menu bar and note down the four numbers that appear below Connectivity. These are your computer's IP address on the network.



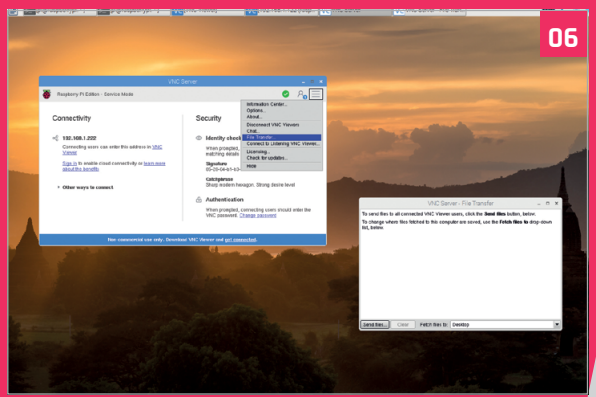
>STEP-03
Open VNC Viewer
If you're connecting from another Raspberry Pi, switch to it, click the Raspberry Pi icon, and select VNC Viewer from the Internet submenu. Enter the IP address of your original machine.



>STEP-04
Optimise performance
If your remote Pi feels sluggish, hover at the top of the VNC Viewer window and click the cog on the menu that appears, then reduce the picture quality on the Options tab.



>STEP-05
Download files
Retrieve files from your remote Pi by clicking the horizontal buttons on the window's drop-down menu, followed by the 'Send files...' button. Choose the files you want and click OK.



>STEP-06
Upload files
To send files to your remote Raspberry Pi, click the menu bar's VNC button, then the menu at the top of the new window. Choose 'File Transfer...' and select the files to transmit.

MIKE'S PI BAKERY

MIKE COOK

Veteran magazine author from the old days and writer of the Body Build series. Co-author of *Raspberry Pi for Dummies*, *Raspberry Pi Projects*, and *Raspberry Pi Projects for Dummies*.
magpi.cc/259aT3X

BUILD AN OSCILLOSCOPE

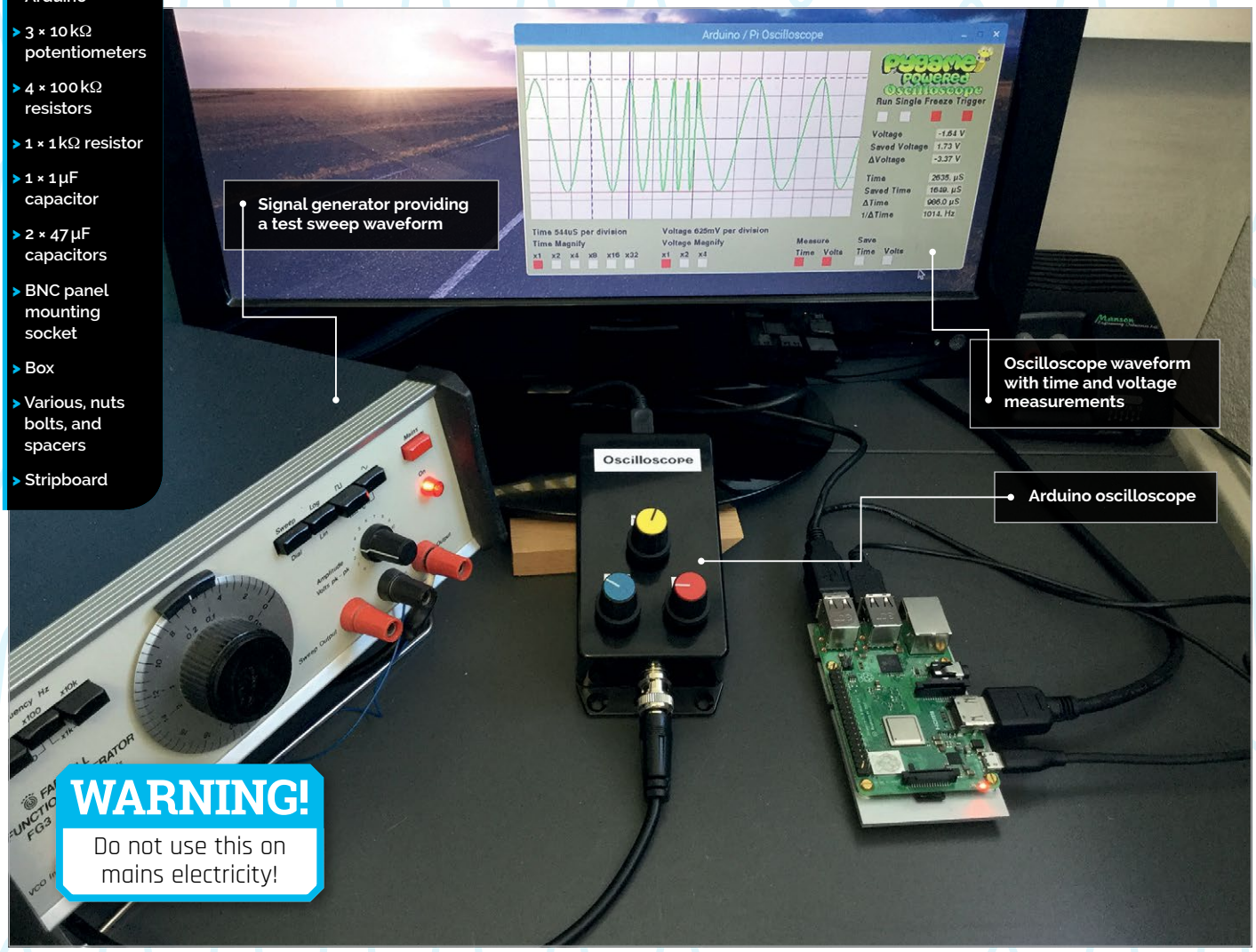
Make your own oscilloscope using a Raspberry Pi and an Arduino

You'll Need

- Uno, Nano, Mega or any AVR-based Arduino
- 3 × 10 kΩ potentiometers
- 4 × 100 kΩ resistors
- 1 × 1 kΩ resistor
- 1 × 1 µF capacitor
- 2 × 47 µF capacitors
- BNC panel mounting socket
- Box
- Various, nuts bolts, and spacers
- Stripboard

The oscilloscope is on the wish list of anyone starting out with electronics. Your author used to tell his students that it was your eyes, making electricity visible. Unfortunately, they are quite expensive: from a few hundred pounds to up to £5000 and beyond. However, by using an Arduino and some software on the Raspberry Pi, you can make a passable beginner's oscilloscope.

Last September, in *The MagPi* #61, there was an article outlining the way the Raspberry Pi and the


Arduino / Pi Oscilloscope
Signal generator providing a test sweep waveform
Oscilloscope waveform with time and voltage measurements
Oscilloscope
Arduino oscilloscope
WARNING!
Do not use this on mains electricity!

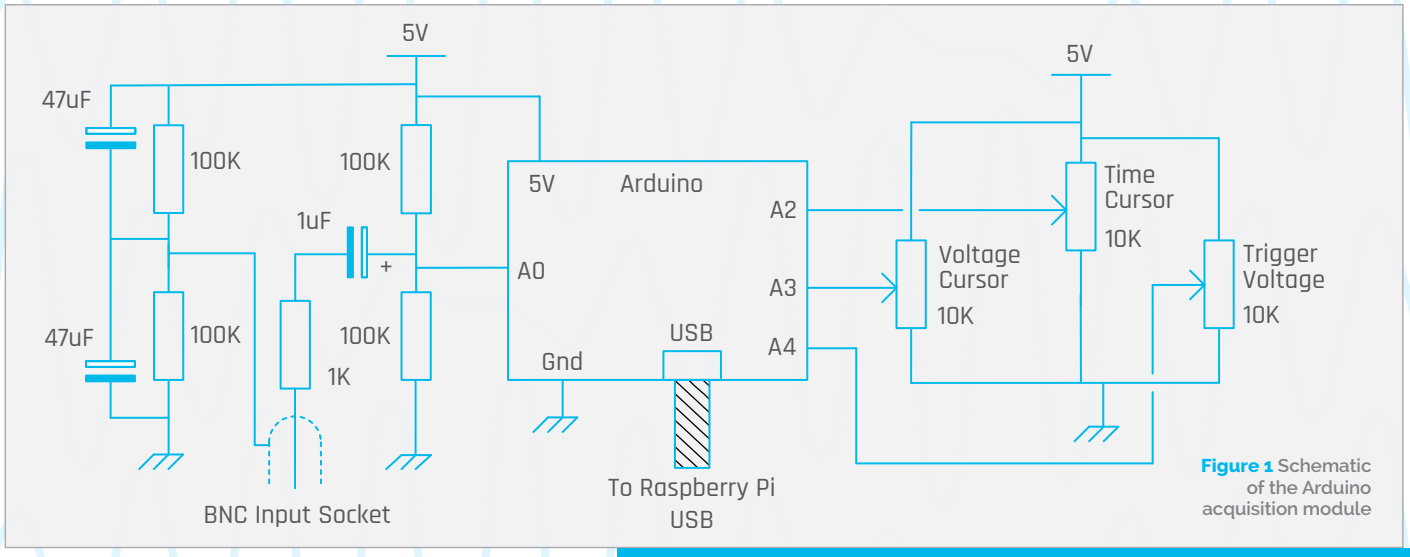


Figure 1 Schematic of the Arduino acquisition module

Arduino could be used together. We at the Bakery have been doing this for some time: we first had a major project in the *Raspberry Pi Projects* books by Andrew Robinson and Mike Cook. The big advantage of the Arduino from a signal processing point of view is that there is minimal interruption from the operating system and you can gather data at a constant uninterrupted rate. This is just what we need for making an oscilloscope. The idea is that the Arduino gathers a whole heap of voltage samples as fast as it can and stores it in memory. It then transfers that memory to the Raspberry Pi, again as fast as possible. The Pi plots the data and displays it, then the software allows measurements to be made on the samples.

So you can measure the time and voltage difference, known as a delta, between any two points on the samples. You can even display the frequency that the 'time delta' corresponds to by taking its reciprocal. These are features found in expensive oscilloscopes. We have also built in a trigger function; this is to synchronise the onset of the rapid data gathering with the occurrence of a positive transition on the input signal through a specified voltage. The result is that regular waveforms can look stable on the display.

The hardware

The schematic of the Arduino data acquisition module is shown in **Figure 1**. You will notice that it is quite simple. It consists of three potentiometers for the oscilloscope's controls and an AC coupled biased voltage input. The capacitor ensures that no DC components from the input get through and gives a modicum of protection for overvoltage. The reference voltage, or ground, is similarly biased as +2.5 V above the Pi's ground level. The use of a BNC socket for the input ensures that you can use this with proper oscilloscope probe leads; these normally have an X10

MAKING THE **DATA ACQUISITION MODULE**

>STEP-01
Gathering the parts

We used an Arduino Nano and soldered the header pins to it. Then we took a 14 hole by 19 strips piece of stripboard and drilled some holes to fix it to the base of the box. You might want to make this longer than 19 strips if you are not using surface-mount resistors on the underside. Fit header sockets to the stripboard and break the tracks on the underside between the two rows.

switchable attenuator fitted, thus allowing voltages of +/- 25 V to be measured. Full construction details can be found in the numbered steps.

Arduino software

The software, or sketch, you need to put into the Arduino is shown in the **Gather_A0.ino** listing, and is quite simple. Normally an Arduino of this type will take samples at a rate of 10 000 per second

> It proved to be a lot more tricky to write than we first imagined

– or as we say, a 10k sample rate. This is not too good for an oscilloscope, but we can increase this sample rate by speeding up the A/D converter's clock speed from the default rate. It does not appear to affect the reading accuracy too much. By making this change, we can speed up the sample rate to 58k. This is much better and allows useful measurements to be made in the audio range.

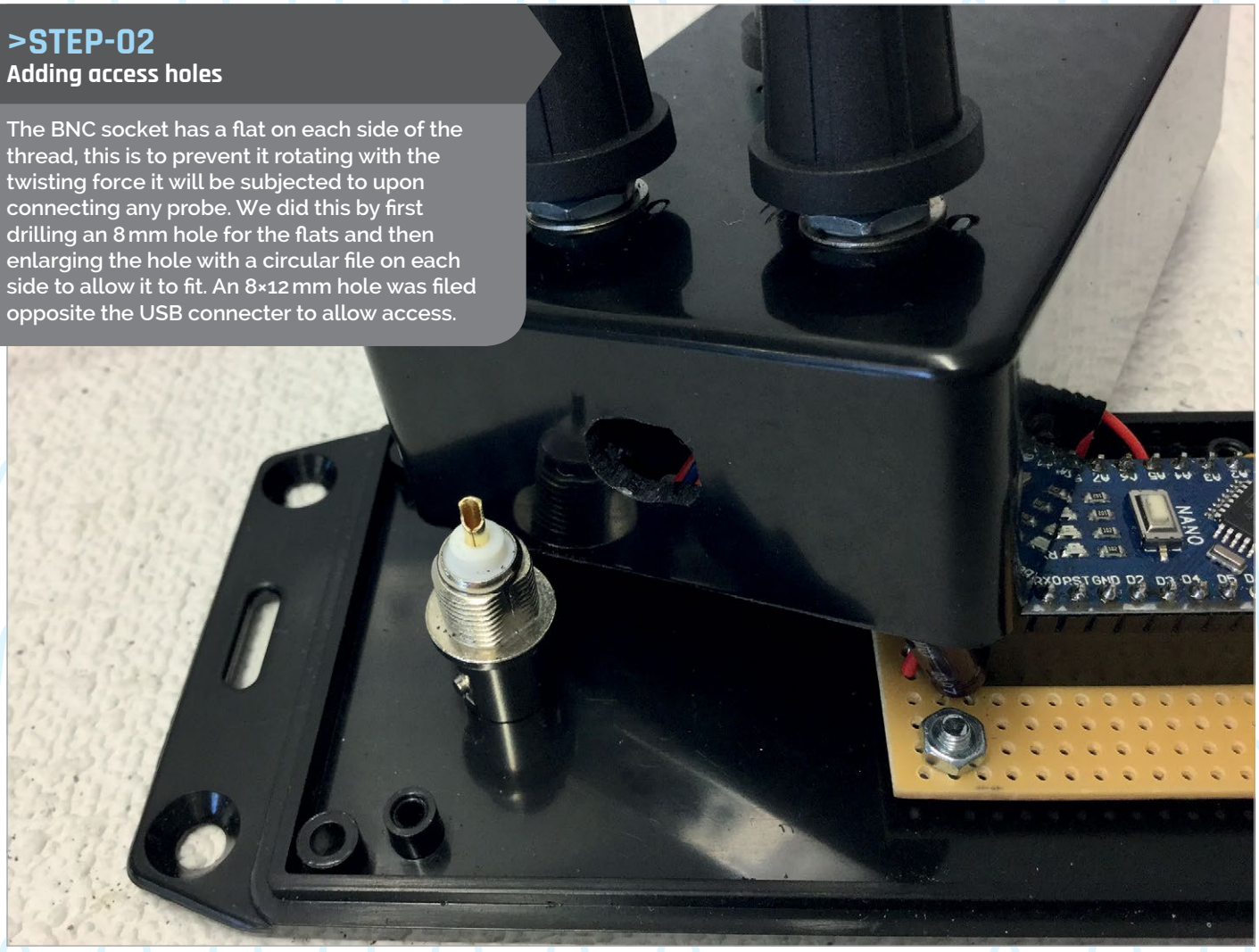
So, first, the trigger function is optionally called and then the samples are gathered in and sent to the Pi. The trigger function has a time-out which means it will trigger anyway after one second, whether it sees a transition on the input signal or not. Then the three pots are measured and also sent to the Pi. Note here that the samples are ten bits wide and so have to be sent as two bytes that get joined together again in the Pi's software.

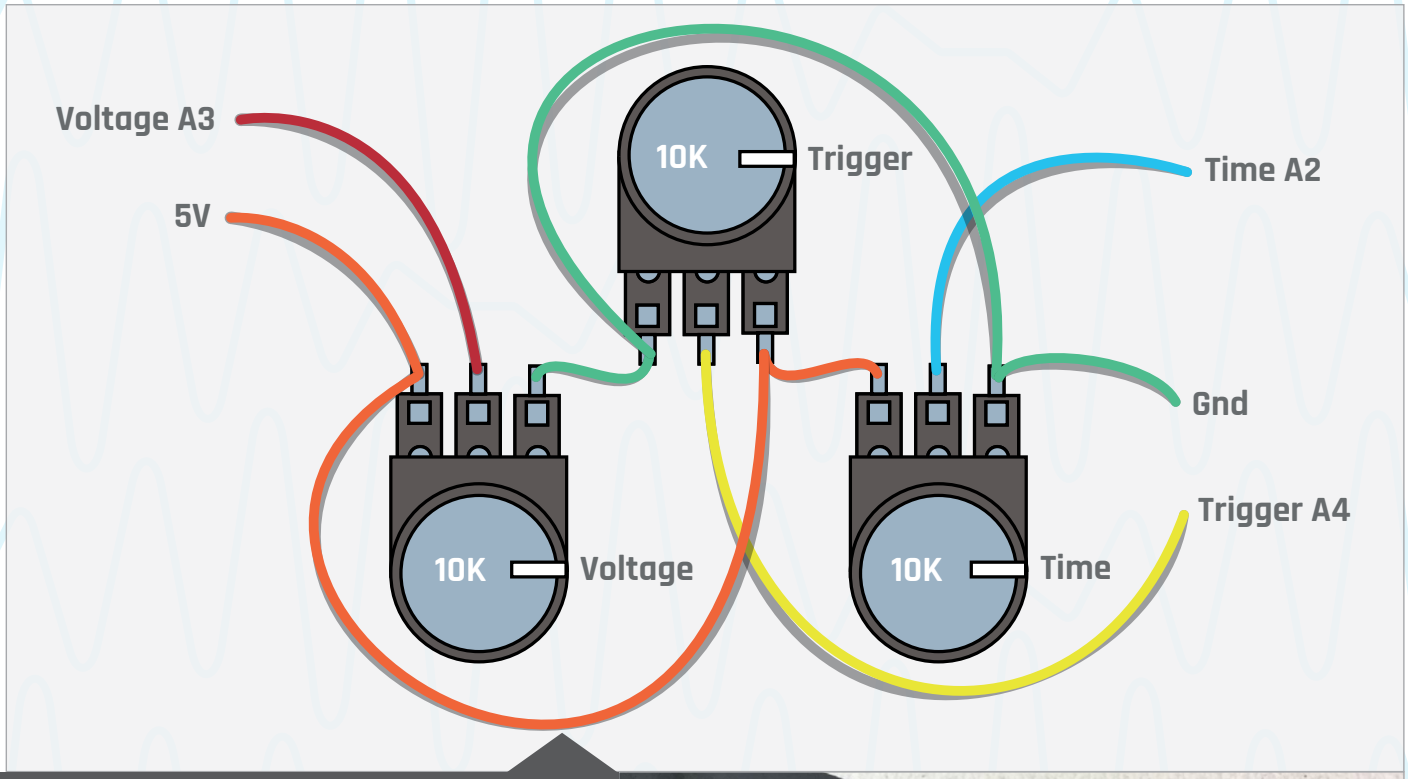
Also note the use of the double read for the pots, with a bit of code between each. This ensures a more stable reading as the input capacitor of the Arduino's sample and hold circuit needs time to charge up, and it has less time than normal to do this due to the speeding up of the D/A. It does not effect the waveform samples too much as in most waveforms one sample voltage is close to the previous one.

At the end of the transfer, the Arduino sits in a loop waiting for an acknowledge byte from the Pi so it can start again. This acknowledge byte also carries

>STEP-02
Adding access holes

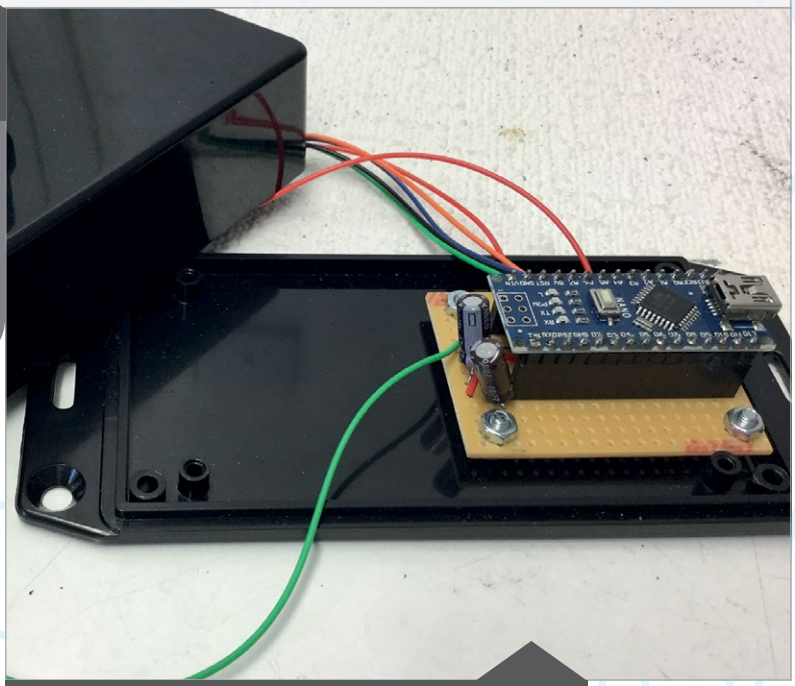
The BNC socket has a flat on each side of the thread, this is to prevent it rotating with the twisting force it will be subjected to upon connecting any probe. We did this by first drilling an 8 mm hole for the flats and then enlarging the hole with a circular file on each side to allow it to fit. An 8×12 mm hole was filed opposite the USB connecter to allow access.





>STEP-03
Wiring the pots

We then drilled three holes for the pots, and added the small slots for the anti-rotation lugs. Then we fitted the pots and wired them up using the diagram above. This is the view from inside the lid of the box; if you're worried about touching the side of the box with your soldering iron, consider soldering them before attaching to the box.



the information as to whether to use a trigger or not on the next sample.

Finally, before each buffer full of data is gathered, pin 13 on the board is lit, and turned off after. This is so that we could time the process on a commercial oscilloscope to find the sample rate – something you will not have to do if you use the recommended AVR-type Arduinos running at 16MHz.

Pi software

The software for the Raspberry Pi is written in Python 3 and used the Pygame framework. It proved to be a lot more tricky to write than we first imagined, and is shown in the **Scope.py** listing. Python 3 uses Unicode characters by default, and allowed us to display the delta (Δ) and mu (µ) Greek characters for the difference and the time. The code first sets up the non-display part of the window, this is only drawn once and then parts of it are updated when necessary. Depending on what type of Arduino you have, it can show up as a different USB port; we found that ours

>STEP-04
Finishing off

Add the resistors and capacitors to the stripboard and wire up the BNC socket. Solder this up before mounting, otherwise you will melt the plastic. Remember to thread the central wire through the ground washer, crinkle washer, and nut before soldering it. Add labels Trigger, Time, and Volts to the knobs.

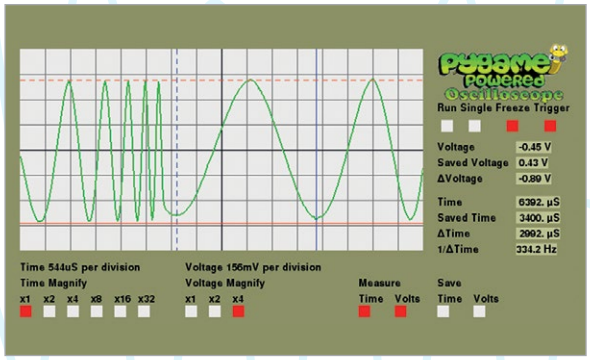


Figure 2 Taking measurements on a swept signal

showed up as one of two ports. Comment out which one is not applicable when defining the **sampleInput** variable at the start of the listing.

Finally, we cobbled together a 168×78 pixel logo for the top-left corner, using a piece of clip art and fashioning the word 'Oscilloscope' from an outlined version of the Cooper Black font. We called it **PyLogo.png** and placed it in an **images** folder next to the Python code.

Using the oscilloscope

The oscilloscope samples at 58 kHz, which in theory means you can measure waveforms at 29 kHz. But that only gives you two samples per cycle and as the samples can be anywhere on the waveform, they do not look very good. As a rough guide, you need at least ten points on a waveform to make it look like a waveform, so that gives a top practical frequency of 5.8 kHz. However, by using the Time Magnify options along with the freeze function, you can measure much higher frequencies. The time and voltage cursor lines let you find out the values on any point of the waveform, and by clicking the save functions the current cursor is replaced by a dotted line which is fixed, and measurements can be made relative to that. The oscilloscope in action can be seen in **Figure 2**. Note that pressing the **S** key on the keyboard produces a screen dump of the display.

Taking it further

There are lots of ways you can take this project further. A simple upgrade would involve you having a second data buffer to allow you to display a saved waveform to compare against the current live one. You could also add a lower speed acquisition mode to see slower waveforms. You can go the other way and use a faster Arduino so you can see the higher frequencies. This oscilloscope is AC coupled; you could add a DC coupling option with a switch potential divider and amplifier to the front end to extend the range of voltages you can measure. All these improvements, however, will need changes to the software to allow the measuring to take place on these wider-range parameters.

Gather_A0.ino

```
001.  int buffer [512]; // 1K input buffer
002.  int sample, lastSample;
003.  int pot1, triggerVoltage;
004.  int triggerTimeout = 1000; // time until auto trigger
005.  unsigned long triggerStart;
006.  char triggerType = '2';
007.
008.  void setup(){
009.    Serial.begin(115200);
010.    pinMode(13,OUTPUT);
011.    // set up fast sampling mode
012.    ADCSRA = (ADCSRA & 0xf8) | 0x04; // set 16 times division
013.  }
014.
015.  void loop(){
016.    if( triggerType != '2') trigger(); // get a trigger
017.    digitalWrite(13,HIGH);// timing marker
018.    for(int i=0; i<512 ; i++){
019.      buffer[i] = analogRead(0);
020.    }
021.    digitalWrite(13,LOW); // timing marker
022.    pot1 = analogRead(2); // switch channel to cursor pot
023.    for(int i=0; i<512 ; i++){
024.      Serial.write(buffer[i]>>8);
025.      Serial.write(buffer[i] & 0xff);
026.    }
027.    // send back pot values for cursors
028.    pot1 = analogRead(2);
029.    analogRead(3); // next cursor pot
030.    Serial.write(pot1>>8);
031.    Serial.write(pot1 & 0xff);
032.    pot1 = analogRead(3);
033.    triggerVoltage = analogRead(4);
034.    Serial.write(pot1>>8);
035.    Serial.write(pot1 & 0xff);
036.    triggerVoltage = analogRead(4);
037.    pot1 = analogRead(0); // prepair for next sample run
038.    Serial.write(triggerVoltage>>8);
039.    Serial.write(triggerVoltage & 0xff);
040.
041.    while(Serial.available() == 0) { } // wait for next request
042.    triggerType = Serial.read(); // see what trigger to use
043.    while (Serial.available() != 0) { // remove any other bytes
      in buffer
044.        Serial.read();
045.    }
046.  }
047.
048.  void trigger(){
049.    // trigger at rising zero crossing
050.    triggerStart = millis();
051.    sample = analogRead(0);
052.    do {
053.    lastSample = sample;
054.    sample = analogRead(0);
055.    }
056.    while(!(lastSample < triggerVoltage && sample
      > triggerVoltage) && (millis() - triggerStart <
      triggerTimeout));
057.  }
```

Scope.py

```
import serial, pygame, os, time

pygame.init()
os.environ['SDL_VIDEO_WINDOW_POS'] = 'center'
pygame.display.set_caption("Arduino / Pi Oscilloscope")
pygame.event.set_allowed(None)
pygame.event.set_allowed([pygame.KEYDOWN,pygame.MOUSEBUTTONDOWN, pygame.QUIT, pygame.MOUSEBUTTONUP])

textHeight=20 ; font = pygame.font.Font(None, textHeight)
screenWidth = 720 ; screenHight = 360
screen = pygame.display.set_mode([screenWidth, screenHight],0,32)
display = pygame.Surface((512,256))
backCol = (150,150,100) ; black = (0,0,0) # background colours
pramCol = (200,200,150) # parameter colour
logo = pygame.image.load("images/PyLogo.png").convert_alpha()

sampleInput = serial.Serial("/dev/ttyUSB0",115200, timeout = 5) # For Mega or nano
#sampleInput = serial.Serial("/dev/ttyACM0",115200, timeout = 5) # For Uno

displayWidth = 512 ; displayHight = 256
LedRect = [ pygame.Rect((0,0),(0,0))]*17
inBuf = [0]*512 # quick way of getting a 512 long buffer
chOff = displayHight//2 # Channel Offset
run = [True,False,False,True,False] # run controls
expandT = 1 ; expandV = 1 # voltage & time expansion

sampleTime = 17 # uS for 58KHz sample
smples_cm = 32 * sampleTime
volts_sample = 5/1024 # volts per sample
measureTime = False ; measureVolts = False;savedTime = 0;savedVoltage = 0
cursorT = 0; cursorV = 0; vMag = 1; svLed = False; stLed = False
triggerC = 512 ; savedVoltsC = -1 ; savedTimeC = -1

def main():
    pygame.draw.rect(screen,backCol,(0,0,screenWidth, screenHight+2),0)
    defineControls()
    drawControls()
    time.sleep(0.1)
    sampleInput.flushInput() # empty any buffer contents
    sampleInput.write(b'2') # tell Arduino to get a new buffer
    while(1):
        time.sleep(0.001) # let other code have a look in
        readArduino() # get buffer data
        plotWave() # draw waveform
        if measureTime or measureVolts :
            updateControls(True)
        drawScope() # display new screen
        checkForEvent()
        while run[4]: # if in hold mode wait here
            checkForEvent()
        if run[3]:
            sampleInput.write(b'1') # tell Arduino to get an other buffers
        else:
            sampleInput.write(b'2') # buffer but no trigger

def drawGrid():
    pygame.draw.rect(display,(240,240,240),(0,0, displayWidth,displayHight),0)
    for h in range(32,256,32): # draw horizontal
        pygame.draw.line(display,(120,120,120),(0,h),(512,h),1)
    for v in range(32,512,32): # draw vertical
        pygame.draw.line(display,(120,120,120),(v,0),(v,256),1)
    pygame.draw.line(display,(0,0,0),(256,0),(256,256),1)
    pygame.draw.line(display,(0,0,0),(0,128),(512,128),1)

def drawControls():
    drawWords("Time Magnify",10,300,black,backCol)
    drawWords("Voltage Magnify",220,300,black,backCol)
    drawWords("Measure",440,300,black,backCol)
    drawWords("Time",440,320,black,backCol)
    drawWords("Volts",486,320,black,backCol)
    drawWords("Save",540,300,black,backCol)
    drawWords("Time",540,320,black,backCol)
    drawWords("Volts",586,320,black,backCol)
    drawWords("1/"+chr(0x394)+"Time",540,257,black, backCol)
    drawWords(chr(0x394)+"Time",540,237,black,backCol)
    drawWords("Saved Time",540,217,black,backCol)
    drawWords("Time",540,197,black,backCol)
    drawWords(chr(0x394)+"Voltage",540,167,black,backCol)
    drawWords("Saved Voltage",540,147,black,backCol)
    drawWords("Voltage",540,127,black,backCol)
    drawWords("Run Single Freeze Trigger", 540,77,black,backCol)
    screen.blit(logo,(540,2))
    updateControls(True)

def updateControls(blank):
    global vDisp
    if blank:
      pygame.draw.rect(screen,backCol,resultsRect,0)
    if expandT*smples_cm >= 1000:
       drawWords("Time "+str((expandT*smples_cm)//1000) +"mS per division    ",10,280,black,backCol)
    else:
        drawWords("Time "+str(expandT*smples_cm)+ "uS per division    ",10,280,black,backCol)
    volts_cm = int(volts_sample*128*1000/expandV)
    drawWords("Voltage "+str(volts_cm)+"mV per division",
```

Language

>C/C++
>PYTHON 3

DOWNLOAD:
magpi.cc/1NqJjmV

PROJECT VIDEOS

Check out Mike's Bakery videos at:
magpi.cc/DsjbZK

```
    220,280,black,backCol)
095.     for n in range(0,6): # time option LED
096.         drawWords("x"+str(1<<n),10+n*30,320,black,
    backCol)
097.         drawLED(n,expandT == 1<<n)
098.     for n in range(6,9): # voltage options
099.         drawWords("x"+str(1<<(n-6)),220+
    (n-6)*30,320,black,backCol)
100.         drawLED(n,expandV == 1<<(n-6))
101.     drawLED(9,measureTime)
102.     drawLED(10,measureVolts)
103.     drawLED(11,stLed)
104.     drawLED(12,svLed)
105.     for n in range(13,17):
106.         drawLED(n,run[n-13])
107.     if measureTime :
108.         t = (cursorT>>1)*sampleTime / expandT
109.         drawWords(" "+trunk(t,5)+" "+chr(0x3bc)+"S",640,
    197,black,pramCol) # current time
110.         drawWords(" "+trunk(savedTime,5)+" "+chr(0x3bc)+
    "S",640,217,black,pramCol)
111.         drawWords(" "+trunk(t-savedTime,5)+" "+chr(0x3bc)
    +"S",640,237,black,pramCol) # delta time
112.         if t-savedTime != 0 :
113.             drawWords((trunk(1000000 / abs(t-savedTime),5))
    +" Hz",640,257,black,pramCol)
114.     if measureVolts :
115.         vDisp = (((1024-cursorV)>>2)-128)*volts_sample *
    vMag
116.         delta = vDisp - savedVoltage
117.         drawWords(" "+trunk(delta,4)+
    "V",640,167,black,pramCol)
118.         drawWords(" "+trunk(savedVoltage,4)+
    "V",640,147,black,pramCol)
119.         drawWords(" "+trunk(vDisp,4)+
    "V",640,127,black,pramCol)
120.
121. def trunk(value, place): # truncate a value string
122.     v=str(value)+"000000"
123.     if value>0:
124.         v = v[0:place]
125.     else:
126.         v = v[0:place+1] # extra place for the minus sign
127.     return v
128.
129. def drawLED(n,state): # draw LED
130.     if state :
131.         pygame.draw.rect(screen,(240,0,0),LedRect[n],0)
132.     else :
133.         pygame.draw.rect(screen,(240,240,240),
    LedRect[n],0)
134.
135. def defineControls():
136.    global LedRect, resultsRect
137.    for n in range(0,6):
138.        LedRect[n] = pygame.Rect((10+n*30,336),(15,15))
139.    for n in range(6,9):
140.        LedRect[n] = pygame.Rect((220+
    (n-6)*30,336),(15,15))
141.    LedRect[9] = pygame.Rect((440,336),(15,15))  # time
142.    LedRect[10] = pygame.Rect((486,336),(15,15)) # volts
143.    LedRect[11] = pygame.Rect((540,336),(15,15)) # save
    time
144.    LedRect[12] = pygame.Rect((586,336),(15,15)) # save
    volts
145.    LedRect[13] = pygame.Rect((545,100),(15,15)) # run
146.    LedRect[14] = pygame.Rect((580,100),(15,15)) # single
147.    LedRect[15] = pygame.Rect((628,100),(15,15)) # freeze
148.    LedRect[16] = pygame.Rect((676,100),(15,15)) #
    trigger
149.    resultsRect = pygame.Rect((639,125),(90,153))
150.
151. def plotWave():
152.     global vMag
153.     lastX=0 ; lastY=0
154.     vMag = 2 # adjust voltage scale
155.     if expandV == 1:
156.         vMag = 4
157.     if expandV == 4:
158.         vMag =1
159.     drawGrid()
160.     s = 0 # sample pointer
161.     for n in range(0, displayWidth, expandT):
162.         y = (512-inBuf[s])//vMag + chOff
163.         if n != 0:
164.             pygame.draw.line(display,(0,200,0),
    (lastX ,lastY), (n ,y ),2)
165.         lastX = n
166.         lastY = y
167.         s += 1
168.     if measureTime :
169.        pygame.draw.line(display,(0,0,255),
    (cursorT>>1,0), (cursorT>>1,256),1)
170.        if savedTimeC != -1:
171.          for n in range(0,256,12):
172.             pygame.draw.line(display,(0,0,255),
    (savedTimeC,n),(savedTimeC,n+6),1)
173.     if measureVolts :
174.        pygame.draw.line(display,(255,0,0),
    (0,cursorV>>2), (512,cursorV>>2),1)
175.        if savedVoltsC != -1:
176.          for n in range(0,512,12):
177.             pygame.draw.line(display,(255,0,0),
    (n,savedVoltsC),(n+6,savedVoltsC),1)
178.     if run[3] : # use trigger
179.        y = (triggerC-512)//vMag + chOff
180.        for n in range(0,512,12):
181.           pygame.draw.line(display,(255,128,0),(n,y),
    (n+6,y),1)
182.
183. def drawScope(): # put display onto scope controls
184.     screen.blit(display,(10,10))
185.     pygame.display.update()
186.
187. def drawWords(words,x,y,col,backCol) :
188.     textSurface = font.render(words, True, col, backCol)
189.     textRect = textSurface.get_rect()
190.     textRect.left = x
```

```
    textRect.top = y
    screen.blit(textSurface, textRect)

def readArduino(): # get buffer and controls
    global cursorT, cursorV, triggerC, run
    if run[2] : #if in freeze mode funnel data into junk
       for i in range(0,1024):
          junk = sampleInput.read()
    else: # otherwise read into the buffer
       for i in range(0,512):
           inBuf[i] = ((ord(sampleInput.read())) << 8) | ord(sampleInput.read())
    cursorT = ((ord(sampleInput.read())) << 8) | ord(sampleInput.read())
    cursorV = 1024 - (((ord(sampleInput.read())) << 8) | ord(sampleInput.read()))
    triggerC = 1024 - (((ord(sampleInput.read())) << 8) | ord(sampleInput.read()))
    if run[1]: #single sweep requested
       run[1] = False
       run[2] = True # put in freeze mode
       updateControls(True)

def handleMouse(pos): # look at mouse down
   global expandT,expandV,measureTime,measureVolts,svLed, stLed
   global savedVoltsC, savedTimeC, run
   #print(pos)
   for n in range(0,6) :
      if LedRect[n].collidepoint(pos):
         expandT = 1<<n
   for n in range(6,9) :
      if LedRect[n].collidepoint(pos):
         expandV = 1<<(n-6)
   if LedRect[9].collidepoint(pos): #toggle time measurement
       measureTime = not(measureTime)
       if not measureTime :
         savedTimeC = -1
   if LedRect[10].collidepoint(pos):
       measureVolts = not(measureVolts) # toggle volts measurement
       if not measureVolts :
           savedVoltsC = -1
   if LedRect[11].collidepoint(pos) and measureTime: # save time
       stLed = True
       savedTimeC = cursorT>>1
   if LedRect[12].collidepoint(pos) and measureVolts: # save volts
       svLed = True
       savedVoltsC = cursorV>>2
   # run controls logic
   if LedRect[13].collidepoint(pos) and not run[1]: # run
       run[0] = not(run[0])
       if not run[0]:
           run[2] = True
       else:
           run[2] = False
   if LedRect[14].collidepoint(pos): # single
         run[1] = True
         run[0] = False
         run[2] = False
         run[4] = True
         updateControls(False)
         drawScope()
   if LedRect[15].collidepoint(pos) and not run[1]: # freeze
       run[2] = not(run[2])
       if not run[2]:
           run[0] = True
       else:
           run[0] = False
   if LedRect[16].collidepoint(pos): # trigger
       run[3] = not(run[3])
   updateControls(False)

def handleMouseUp(pos): # look at mouse up
   global savedVoltage,savedTime, svLed, stLed, run
   if LedRect[12].collidepoint(pos) and measureVolts:
       savedVoltage = vDisp
       svLed = False
       updateControls(False)
   if LedRect[11].collidepoint(pos) and measureTime:
       savedTime = (cursorT>>1)*sampleTime / expandT
       stLed = False
       updateControls(False)
   if LedRect[14].collidepoint(pos): # single
       run[4] = False
       updateControls(False)

def terminate(): # close down the program
    pygame.quit() # close pygame
    os._exit(1)

def checkForEvent(): # see if we need to quit
    event = pygame.event.poll()
    if event.type == pygame.QUIT :
         terminate()
    if event.type == pygame.KEYDOWN :
       if event.key == pygame.K_ESCAPE :
          terminate()
       if event.key == pygame.K_s : # screen dump
          os.system("scrot -u")
    if event.type == pygame.MOUSEBUTTONDOWN :
        handleMouse(pygame.mouse.get_pos())
    if event.type == pygame.MOUSEBUTTONUP :
        handleMouseUp(pygame.mouse.get_pos())


# Main program logic:
if __name__ == '__main__':
    main()
```

MARK VANSTONE

Educational software author from the nineties, author of the ArcVenture series, disappeared into the corporate software wasteland. Rescued by the Raspberry Pi!
technovisualeducation.co.uk
twitter.com/mindexplorers

GET STARTED WITH PYGAME ZERO

You'll Need

- Raspbian Jessie or newer
- An image manipulation program such as GIMP
- A little imagination
- A keyboard

Pygame Zero is a great choice for anyone who wants to start writing computer games on the Raspberry Pi

If you've done some Python coding and wanted to write a game, you may have come across Pygame. The Pygame module adds many functions that help you to write games in Python. Pygame Zero goes one step further to let you skip over the cumbersome process of making all those game loops and setting up your program structure. You don't need to worry about functions to load graphics or keeping data structures for all the game elements. If you just want to get stuck in and start making things happen on the screen without all the fluff, then Pygame Zero is what you need.

>STEP-01

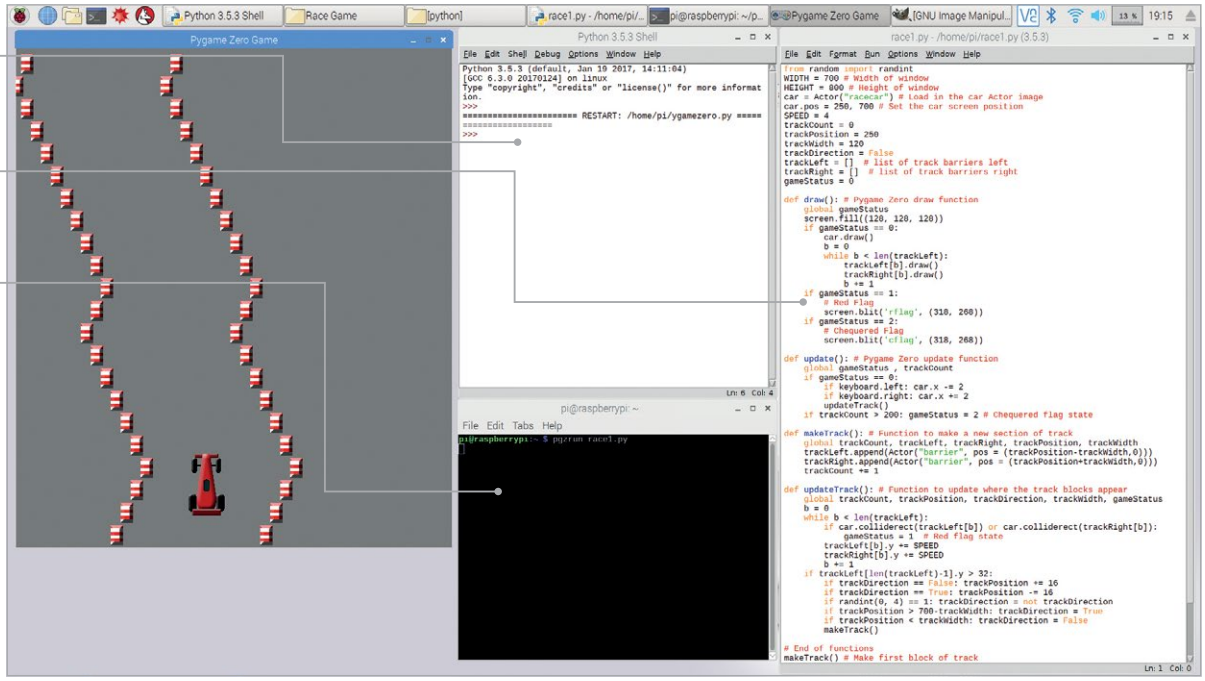
Loading a suitable program editor

The first really labour-saving thing about Pygame Zero is that you can write a program in a simple text editor. For the easiest route we suggest using the IDLE Python 3 editor, as Pygame Zero needs to be formatted like Python with its indents and you'll get the benefit of syntax highlighting to help you along the way.
So the first step in your journey will be to open the Python 3 IDLE editor from the Raspbian main menu, under Programming. You'll be presented with the Python Shell window.

The Python Shell window that appears when we open IDLE

Our program listing. This is a file window from the IDLE application

The Terminal window – enter the command to run our program



TERMINAL SHORTCUTS

Instead of retyping `pgzrun race1.py` in the Terminal window, you can use the up arrow to repeat the last command.

>STEP-02

Writing a Pygame Zero program

To start writing your first Pygame Zero program, go to the File menu of the IDLE Python Shell window and select 'New File' to open up a new editor window – and that's it! You have written your first Pygame Zero program! The Pygame Zero framework assumes that you will want to open a new window to run your game inside, so even a blank file will create a running game environment. Of course at this stage your game doesn't do very much, but you can test it to make sure that you can get a program running.

>STEP-03

Running your first Pygame Zero program

With other Python programs, you can run them directly from the Python file window. Currently IDLE does not support running Pygame Zero programs directly, but the alternative is very simple. First of all, you need to save your blank program file. We suggest saving it as **pygame1.py** in your default user folder (just save the file without changing directory). All you need to do then is open a Terminal window from the main Raspbian menu and type **`pgzrun pygame1.py`** (assuming you called your program **pygame1.py**) and hit **RETURN**. After a few seconds, a window titled 'Pygame Zero Game' should appear.

>STEP-04

Setting up the basics

By default, the Pygame Zero window opens at the size of 800 pixels wide by 600 pixels high. If you want to change the size of your window, there are two predefined variables you can set. If you include **`WIDTH = 700`** in your program, then the window will be set at 700 pixels wide. If you include **`HEIGHT = 800`**, then the window will be set to 800 pixels high. In this tutorial we'll be writing a simple racing game, so we want our window to be a bit taller than it is wide. When you have set the **`WIDTH`** and **`HEIGHT`** variables, you could save your file as **race1.py** and test it like before by typing **`pgzrun race1.py`** into the Terminal window.

>STEP-05

Look! No game loop!

When writing a Python game, normally you would have a game loop – that's a piece of code that is run over and over while the game is running. Pygame Zero does away with this idea and provides predefined functions to handle each of the tasks that the game loop normally performs. The first of these we will look at is the function **`draw()`**. We can write this function into our program the same as we would normally define a function in Python, which is **`def draw():`**. Then, so that you can see the draw function doing something, add a line underneath indented by one tab: **`screen.fill((128, 128, 128))`**. This is shown in the **figure1.py** listing overleaf.

race1.py

Language
>PYTHON

DOWNLOAD:
magpi.cc/VcqutR

```
01. from random import randint
02. WIDTH = 700 # Width of window
03. HEIGHT = 800 # Height of window
04. car = Actor("racecar") # Load in the
    car Actor image
05. car.pos = 250, 700 # Set the car screen position
06. SPEED = 4
07. trackCount = 0
08. trackPosition = 250
09. trackWidth = 120
10. trackDirection = False
11. trackLeft = []  # list of track barriers left
12. trackRight = []  # list of track barriers right
13. gameStatus = 0
14.
15. def draw(): # Pygame Zero draw function
16.     global gameStatus
17.     screen.fill((128, 128, 128))
18.     if gameStatus == 0:
19.         car.draw()
20.         b = 0
21.         while b < len(trackLeft):
22.             trackLeft[b].draw()
23.             trackRight[b].draw()
24.             b += 1
25.     if gameStatus == 1:
26.         # Red Flag
27.         screen.blit('rflag', (318, 268))
28.     if gameStatus == 2:
29.         # Chequered Flag
30.         screen.blit('cflag', (318, 268))
31.
32. def update(): # Pygame Zero update function
33.     global gameStatus , trackCount
34.     if gameStatus == 0:
35.         if keyboard.left: car.x -= 2
36.         if keyboard.right: car.x += 2
37.         updateTrack()
38.     if trackCount > 200: gameStatus = 2 # Chequered flag
    state
39.
40. def makeTrack(): # Function to make a new section of track
41.     global trackCount, trackLeft, trackRight, trackPosition,
    trackWidth
42.     trackLeft.append(Actor("barrier", pos = (trackPosition-
    trackWidth,0)))
43.     trackRight.append(Actor("barrier", pos =
    (trackPosition+trackWidth,0)))
44.     trackCount += 1
45.
46. def updateTrack(): # Function to update where the track
    blocks appear
47.     global trackCount, trackPosition, trackDirection,
    trackWidth, gameStatus
48.     b = 0
49.     while b < len(trackLeft):
50.         if car.colliderect(trackLeft[b]) or
    car.colliderect(trackRight[b]):
51.             gameStatus = 1  # Red flag state
52.         trackLeft[b].y += SPEED
53.         trackRight[b].y += SPEED
54.         b += 1
55.     if trackLeft[len(trackLeft)-1].y > 32:
56.         if trackDirection == False: trackPosition += 16
57.         if trackDirection == True: trackPosition -= 16
58.         if randint(0, 4) == 1: trackDirection = not
    trackDirection
59.         if trackPosition > 700-trackWidth: trackDirection =
    True
60.         if trackPosition < trackWidth: trackDirection =
    False
61.         makeTrack()
62.
63. # End of functions
64. makeTrack() # Make first block of track
```

figure1.py

```
01. WIDTH = 700
02. HEIGHT = 800
03.
04. def draw():
05.     screen.fill((128, 128, 128))
```

Figure 1 To set the height and width of a Pygame Zero window, just set the variables `HEIGHT` and `WIDTH`. Then you can fill the screen with a colour

>STEP-06

The Python format

You may have noticed that in the previous step we said to indent the **`screen.fill`** line by one tab. Pygame Zero follows the same formatting rules as Python, so you will need to take care to indent your code correctly. The indents in Python show that the code is inside a structure. So if you define a function, all the code inside it will be indented by one tab. If you then have a condition or a loop, for example an **`if`** statement, then the contents of that condition will be indented by another tab (so two in total).

> Actors in Pygame Zero are dynamic graphic objects, much the same as sprites

>STEP-07

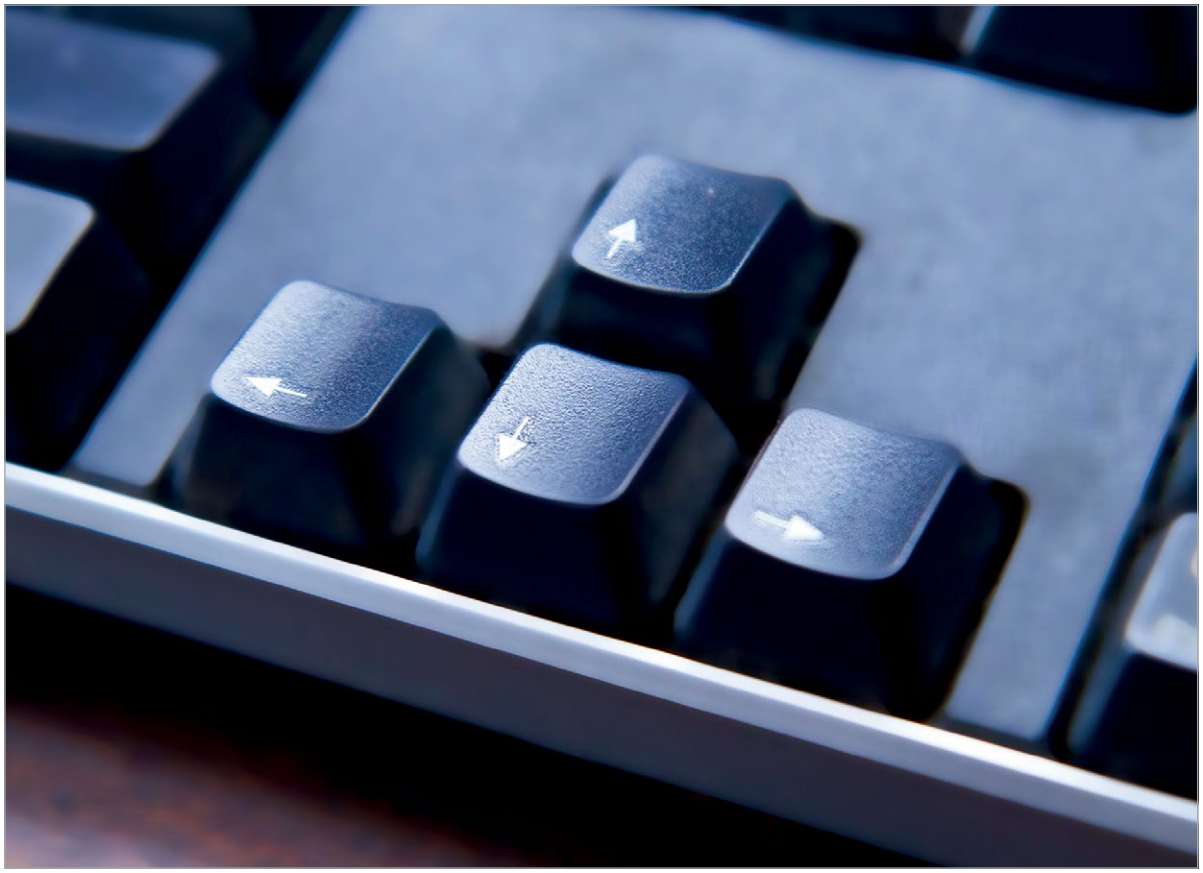
All the world's a stage

The screen object used in Step 5 is a predefined object that refers to the window we've opened for our game. The **`fill`** function fills the window with the RGB value (a tuple value) provided – in this case, a shade of grey. Now that we have our stage set, we can create our Actors. Actors in Pygame Zero are dynamic graphic objects, much the same as sprites in other programming systems. We can load an Actor by typing **`car = Actor("racecar")`**. This is best placed near the top of your program, before the **`draw()`** function.

>STEP-08

It's all about image

When we define an Actor in our program, what we are actually doing is saying 'go and get this image'. In Pygame Zero our images need to be stored in a directory called **images**, next to our program file. So our Actor would be looking for an image file in the **images** folder called **racecar.png**. It could be a GIF or a JPG file, but it is recommended that your images are PNG files as that file type provides good-quality images with transparencies. You can get a full free image creation program called GIMP by typing **`sudo apt-get install gimp`** in your Terminal window. If you want to use our images, you can download them from **magpi.cc/srHWWH**.



Right To respond to key presses, Pygame Zero has a built-in object called `keyboard`. The arrow key states can be read with `keyboard.up`, `keyboard.down`, and so on

THE GRAPHICS

If you use PNG files for your graphics rather than JPGs, you can keep part of the image transparent.

>STEP-09

Drawing your Actor

Once you have loaded in your image by defining your Actor, you can set its position on the screen. You can do this straight after loading the Actor by typing **`car.pos = 250, 500`** to set it at position 250, 500 on the screen. Now, when the **`draw()`** function runs, we want to display our race car at the co-ordinates that we have set. So, in our **`draw()`** function, after the **`screen.fill`** command we can type **`car.draw()`**. This will draw our race car at its defined position. Test your program to make sure this is working, by saving it and running **`pgzrun race1.py`** as before.

>STEP-10

I'm a control freak!

Once we have our car drawing on the screen, the next stage is to enable the player to move it backwards and forwards. We can do this with key presses; in this case we are going to use the left and right arrow keys. We can read the state of these keys inside another predefined function called **`update()`**. We can type in the definition of this function by adding **`def update():`** to our program. This function is continually checked while the game is running. We can now add an indented **`if`** statement to check the state of a key; e.g., **`if keyboard.left:`**.

>STEP-11

Steering the car

We need to write some code to detect key presses of both arrow keys and also to do something about it if we detect that either has been pressed. Continuing from our **`if keyboard.left:`** line, we can write **`car.x -= 2`**. This means subtract 2 from the car's x co-ordinate. It could also be written in long-hand as **`car.x = car.x - 2`**. Then, on the next line and with the same indent as the first **`if`** statement, we can do the same for the right arrow; i.e., **`if keyboard.right: car.x += 2`**. These lines of code will move the car Actor left and right.

>STEP-12

The long and winding road

Now that we have a car that we can steer, we need a track for it to drive on. We are going to build our track out of Actors, one row at a time. We will need to make some lists to keep track of the Actors we create. To create our lists, we can write the following near the top of our program: **`trackLeft = []`** (note the square brackets) and then, on the next line, **`trackRight = []`**. This creates two empty lists: one to hold the data about the left side of the track, and one to hold the data about the right-hand side.

figure2.py

```
01. def makeTrack(): # Function to make a new section of track
02.     global trackCount, trackLeft, trackRight,
    trackPosition, trackWidth
03.     trackLeft.append(Actor("barrier", pos =
    (trackPosition-trackWidth,0)))
04.     trackRight.append(Actor("barrier", pos =
    (trackPosition+trackWidth,0)))
05.     trackCount += 1
```

>STEP-13

Building the track

We will need to set up a few more variables for the track. After your two lists, declare the following variables: **`trackCount = 0`** and then **`trackPosition = 250`**, then **`trackWidth = 120`**, and finally **`trackDirection = false`**. Then let's make a new function called **`makeTrack()`**. Define this function after your **`update()`** function. See the **figure2.py** listing for the code to put inside **`makeTrack()`**. The function will add one track Actor on the left and one on the right, both using the image **barrier.png**. Each time we call this function, it will add a section of track at the top of the screen.

Figure 2 The `makeTrack()` function. This creates two new Actors with the barrier image at the top of the screen

>STEP-14

On the move

The next thing that we need to do is to move the sections of track down the screen towards the car. Let's write a new function called **`updateTrack()`**. We will call this function in our **`update()`** function after

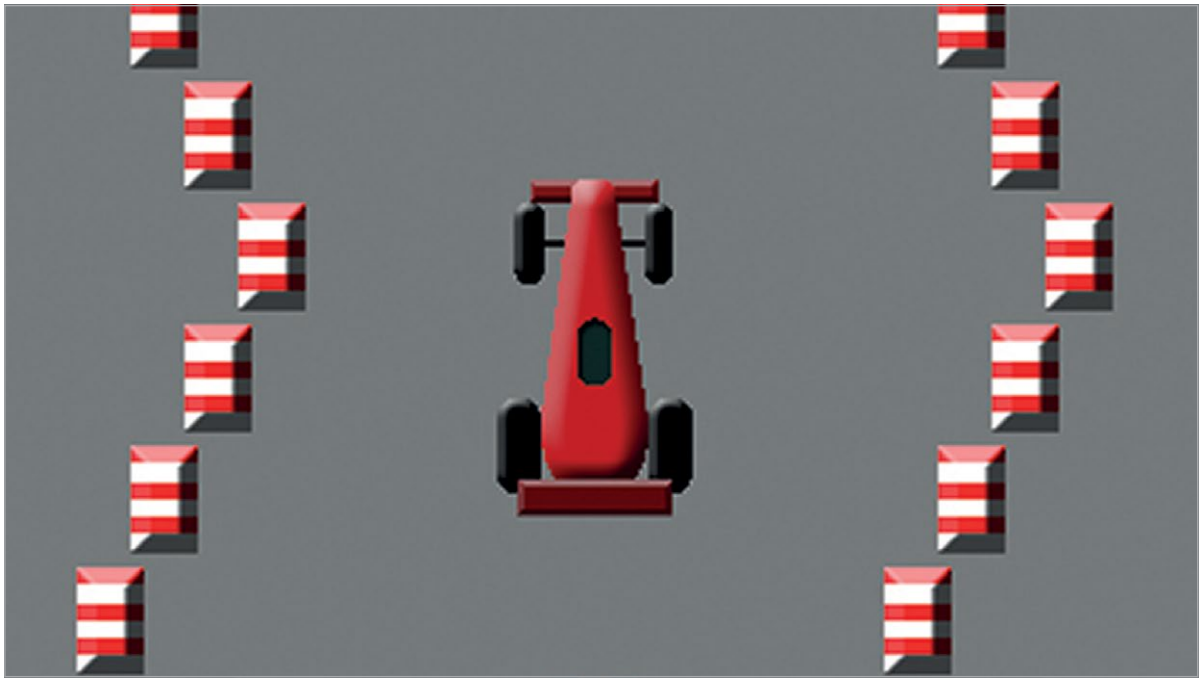
Figure 3 The `updateTrack()` function. Notice the constant SPEED – we need to define this at the top of our program, perhaps starting with the value 4

figure3.py

```
01. def updateTrack(): # Function to update where the track
    blocks appear
02.     global trackCount, trackPosition, trackDirection,
    trackWidth
03.     b = 0
04.     while b < len(trackLeft):
05.         trackLeft[b].y += SPEED
06.         trackRight[b].y += SPEED
07.         b += 1
08.     if trackLeft[len(trackLeft)-1].y > 32:
09.         if trackDirection == False: trackPosition += 16
10.         if trackDirection == True: trackPosition -= 16
11.         if randint(0, 4) == 1: trackDirection = not
    trackDirection
12.         if trackPosition > 700-trackWidth:
    trackDirection = True
13.         if trackPosition < trackWidth: trackDirection =
    False
14.         makeTrack()
```

CHANGING THE TRACK WIDTH

You can make the game easier or harder by changing the `trackWidth` variable to make the track a different width.



Right The race car with barriers making up a track to stay within. The track pieces are created by random numbers so each play is different

Figure 4 The `draw()` function and the `update()` function with conditions (`if` statements) to do different things depending on the value of `gameStatus`

we do the keyboard checks. See the **figure3.py** listing for the code for our **`updateTrack()`** function. In this function we are using **`randint()`**. This is a function that we must load from an external module, so at the top of our code we write **`from random import randint`**. We use this function to make the track curve backwards and forwards.

figure4.py

```
01. def draw(): # Pygame Zero draw function
02.     global gameStatus
03.     screen.fill((128, 128, 128))
04.     if gameStatus == 0:
05.         car.draw()
06.         b = 0
07.         while b < len(trackLeft):
08.             trackLeft[b].draw()
09.             trackRight[b].draw()
10.             b += 1
11.     if gameStatus == 1:
12.         # Red Flag
13.
14.     if gameStatus == 2:
15.         # Chequered Flag
16.
17. def update(): # Pygame Zero update function
18.     global gameStatus , trackCount
19.     if gameStatus == 0:
20.         if keyboard.left: car.x -= 2
21.         if keyboard.right: car.x += 2
22.         updateTrack()
23.     if trackCount > 200: gameStatus = 2 # Chequered
    flag state
```

>STEP-15

Making more track

Notice at the bottom of the **`updateTrack()`** function, there is a call to our **`makeTrack()`** function. This means that for each update when the track sections move down, a new track section is created at the top of the screen. We will need to start this process off, so we will put a call to **`makeTrack()`** at the bottom of our code. If we run our code at the moment, we should see a track snaking down towards the car. The only problem is that we can move the car over the track barriers and we want to keep the car inside them with some collision detection.

>STEP-16

A bit of a car crash

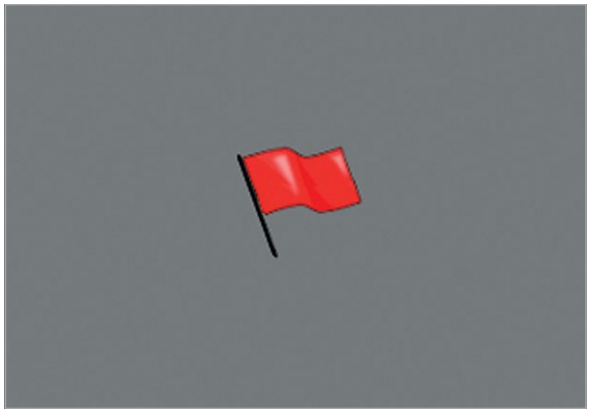
We need to make sure that our car doesn't touch the track Actors. As we are looking through the existing barrier Actors in our **`updateTrack()`** function, we may as well test for collisions at the same time. We can write **`if car.colliderect(trackLeft[b]) or car.colliderect(trackRight[b]):`** and then, indented on the next line, **`gameStatus = 1`**. We have not covered **`gameStatus`** yet – we are going to use this variable to show if the game is running, the car has crashed, or we have reached the end of the race. Define your **`gameStatus`** variable near the top of the program as **`gameStatus = 0`**. You will also need to add it to the global variables in the **`updateTrack()`** function.

>STEP-17

Changing state

In this game we will have three different states to the game stored in our variable **`gameStatus`**. The first or default state will be that the game is running and will be represented by the number 0. The next state will

Below Each of the barrier blocks is checked against the car to detect collisions. If the car hits a barrier, the red flag graphic is displayed



be set if the car crashes, which will be the number 1. The third state will be if we have finished the race, which we'll set as the number 2 in **gameStatus**. We will need to reorganise our **draw()** function and our **update()** function to respond to the **gameStatus** variable. See the **figure4.py** listing for how we do that.

>STEP-18

Finishing touches

All we need to do now is to display something if **gameStatus** is set to 1 or 2. If **gameStatus** is 1 then it means that the car has crashed and we should display a red flag. We can do that with the code:

screen.blit('rflag', (318, 268)). To see if the car has reached the finish, we should count how many track sections have been created and then perhaps when we get to 200, set **gameStatus** to 2. We can do this in the **update()** function as in **Figure 4**. Then, in the **draw()** function, if the **gameStatus** is 2, then we can write **screen.blit('cflag', (318, 268))**. Have a look at the full code listing to see how this all fits together.

> The next thing that we need to do is to move the sections of track down the screen

>STEP-19

Did you win?

If you didn't get the program working first time, you are not alone – it's quite rare to have everything exactly right first time. Check that you have written all the variables and functions correctly and that the capital letters are in the right places. Python also insists on having code properly formatted with indents. When it's all in place, test your program as before and you should have a racing game with a chequered flag at the end!

CHANGING THE SPEED

If you want to make the track move faster or slower, try changing the value of **SPEED** at the start of the program.

The official Pygame Zero documentation can be found at magpi.cc/fBqznh

LUCY HATTERSLEY

Lucy is the editor of *The MagPi* magazine and a general coding spod. We all know she's secretly trying to build Twiki from *Buck Rogers*.
magpi.cc

You'll Need

- Raspbian
- 1GB USB stick
- 16GB microSD card
- TensorFlow

USE TENSORFLOW ON RASPBERRY PI

Discover how to install and use Google's TensorFlow framework to learn AI techniques and add AI to your future projects

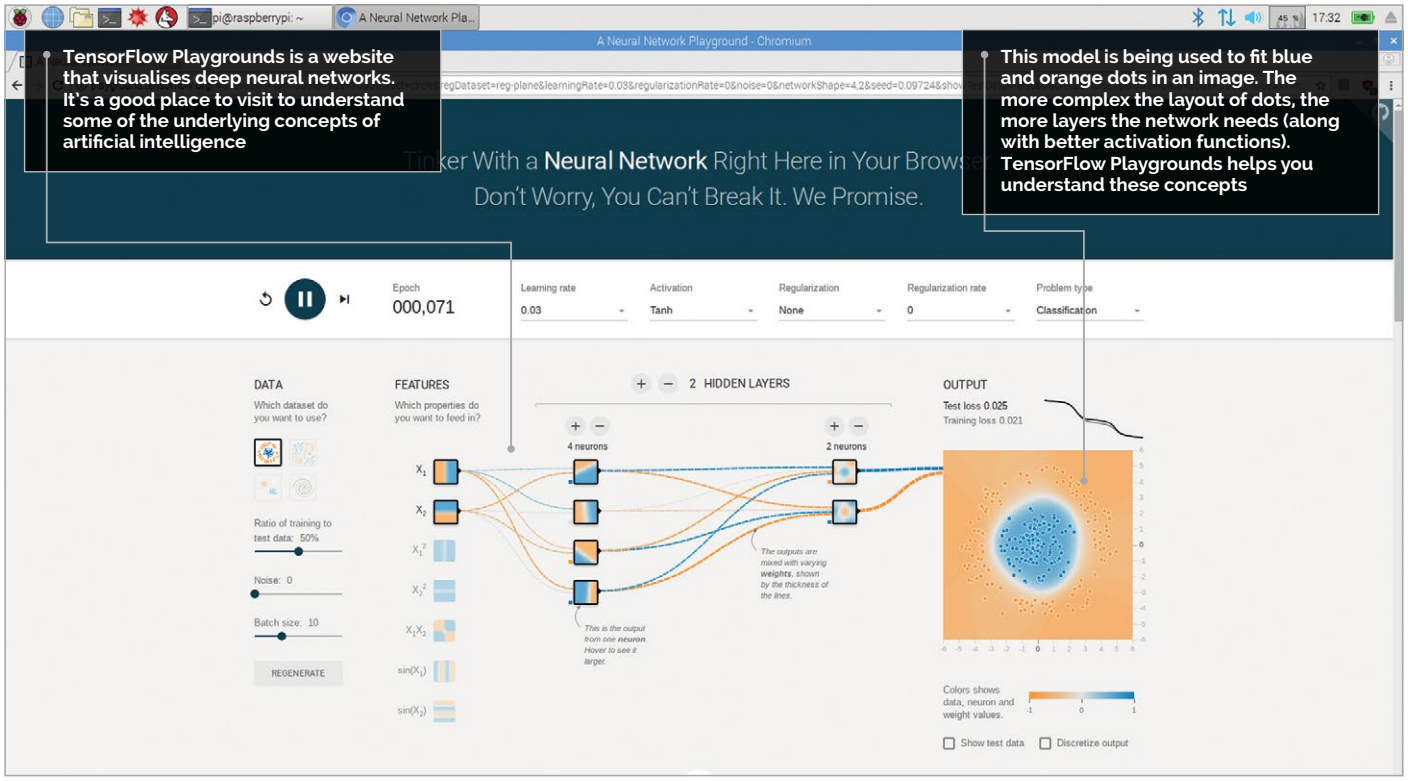
Google TensorFlow is a powerful open-source software framework used to power AI projects around the globe.

TensorFlow is used for machine learning and the creation of neural networks. These make it possible for computers to perform increasingly complex tasks, such as image recognition and text analysis.

When it comes to AI, most people think of powerful supercomputers crunching billions of numbers in giant databanks. But there are two parts to machine learning. There is a train/test part, where you use a lot of data to build a model. And there's deployment, where you take a model and use it as part of a project. And that's where the Raspberry Pi fits in.

Although Raspberry Pi isn't officially supported by Google, there are example models included for the Raspberry Pi and it can be fun (if a bit hacky) to get TensorFlow up and running on a Pi. And there are lots of interesting community projects around that put TensorFlow to good use.

Using TensorFlow can give you a good understanding of how AI works, and how to put AI to practical use in your projects.


TensorFlow Playgrounds is a website that visualises deep neural networks. It's a good place to visit to understand some of the underlying concepts of artificial intelligence
This model is being used to fit blue and orange dots in an image. The more complex the layout of dots, the more layers the network needs (along with better activation functions). TensorFlow Playgrounds helps you understand these concepts
A Neural Network Playground - Chromium
Don't Worry, You Can't Break It. We Promise.
Epoch
000,071
Learning rate
0.03
Activation
Tanh
Regularization
None
Regularization rate
0
Problem type
Classification
DATA
FEATURES
2 HIDDEN LAYERS
4 neurons
2 neurons
OUTPUT
Test loss 0.025
Training loss 0.021
REGENERATE

>STEP-01

Install TensorFlow with pip

TensorFlow can be incredibly easy to install on a Raspberry Pi, or something of a nightmare. It depends on the current build and which version of Raspbian OS you are running. Installation is often troublesome, but we've had recent success with building it directly using pip. Open a Terminal window and enter:

```
sudo apt-get update && sudo apt-get upgrade
sudo apt-get install python3-pip python3-dev
pip3 install tensorflow
```

>STEP-02

Build from wheel

If pip doesn't work, you can try to build TensorFlow using the wheel file. In a Terminal, enter:

```
sudo pip3 install --upgrade https://
storage.googleapis.com/tensorflow/linux/cpu/
tensorflow-1.8.0-cp34-cp34m-linux_x86_64.whl
```

Alternatively, you can use a nightly wheel built for Raspberry Pi, which is available from **magpi.cc/xKLBzu**. Download the wheel file and run it, like this:

```
sudo pip3 install --upgrade tensorflow-
1.9.0rc0-cp34-none-linux_armv7l.whl
```

Take a look at TensorFlow's Install Sources page (**magpi.cc/yIpbCX**) or Common Installation Problems page (**magpi.cc/EHocYB**).

>STEP-03

Build from source

If pip fails, you could always build TensorFlow from source; Sam Abrahams has written detailed instructions on doing so (**magpi.cc/oCYtme**). You will need a spare USB stick (1GB or higher) to extend the amount of swap space on your Raspberry Pi and be sure to follow the instructions carefully. It takes around six hours to build, but we have gone through the steps and they do work.

>STEP-04

Hello TensorFlow

Hopefully, you now have TensorFlow up and running. So let's start it up. Open Python 3 (IDLE) using Menu > Programming > Python 3 (IDLE). Choose File > New File and enter the **hello_tensorflow.py** code.

Save the code file as **hello_tensorflow.py** and Choose Run > Run Module. You will get a warning because TensorFlow is compiled for Python 3.4 and we're running Python 3.5. Don't worry, the code works. The Python shell will display:

```
'Hello TensorFlow'
```

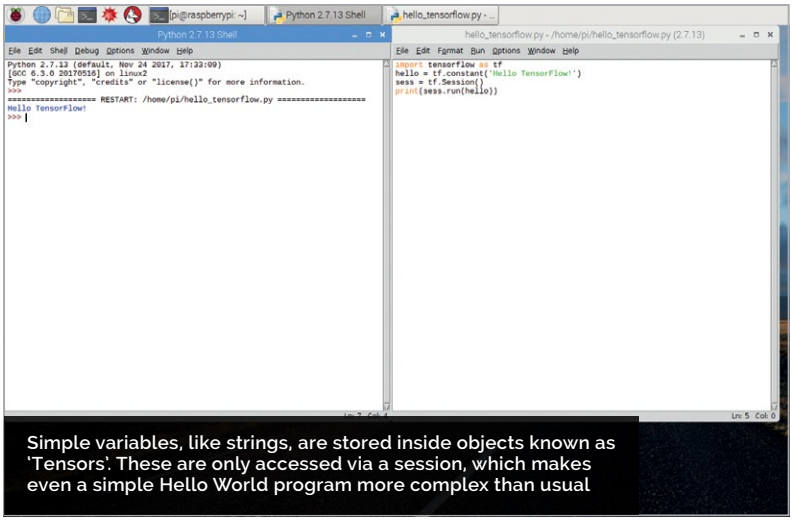
hello_tensorflow.py

```
import tensorflow as tf
hello = tf.constant('Hello, TensorFlow!')
sess = tf.Session()
print(sess.run(hello))
```

Language

>PYTHON

DOWNLOAD:
magpi.cc/HuGfDm



Simple variables, like strings, are stored inside objects known as 'Tensors'. These are only accessed via a session, which makes even a simple Hello World program more complex than usual

>STEP-05

Pi examples

Google has a bunch of models developed for Raspberry Pi that you can test out. Start by cloning the TensorFlow repository:

```
git clone https://github.com/tensorflow/
tensorflow.git
```

Follow the instructions from on the page at **magpi.cc/BrsbKi** to build the example models.

Now head to the part of the TensorFlow repository at **magpi.cc/sGOzbr** to find Google example models and instructions.

The default example is a picture of Grace Hopper. Run it and you will see that it identifies a 'military uniform', 'suit', and 'academic gown' (and then other items in order of decreasing probability). From here you can see how this model could be used to identify objects in your own images, and use that in your own code. There is also a link to an example that uses the Pi Camera Module directly: **magpi.cc/xGurWt**.

>STEP-06

Community TensorFlow

Now you have everything you need to start using TensorFlow. It's a big subject and there's far more to it than we could outline in this tutorial (or even this entire magazine). Learn by doing and follow some TensorFlow projects. Start with Sarthak Jain's 'How to easily detect objects with deep learning on Raspberry Pi' (**magpi.cc/DFFAYt**) or Alasdair Allan's 'Magic mirror with TensorFlow' (**magpi.cc/YGtrOB**).

SIGN UP FOR THIS UDEMY COURSE

If you want to learn TensorFlow in depth, or are finding the 'TensorFlow without a PhD' course limiting, then sign up for a course. There are lots of different courses out there, and we've tried a lot of them. Jose Portilla is the Head of Data Science at Pierian Data and his Udemy course 'Complete Guide to TensorFlow' (**magpi.cc/VJUtKJ**) is the best TensorFlow course we have found.

BRIAN BEUKEN

Very old game programmer now teaching very young game programmers a lot of bad habits at Breda University of Applied Science in Breda NL.
scratchpadgames.net

CODING GAMES ON THE RASPBERRY PI IN C/C++ PART 07

Animate a character and start making a platform game

You'll Need

- Code::Blocks `sudo apt-get codeblocks`
- stb_image.h

We've gone as far as we can with the basic bat-and-ball game, so it's time to introduce another stalwart of the 2D era and create a simple platform game – this time with a character we can control, who can walk, jump, and climb, to avoid some fairly unintelligent baddies.

Animation 101

What is animation? It might seem obvious, but why is it that cartoons and movies allow us to see multiple individual frames as smooth(ish) motion?

All screen-based visuals are based on a simple concept called image retention, where we trick the brain into thinking that a sequence of rapidly changing frames (each not massively different from the previous one) appear to be in motion. As long as we can keep the frame rate up, these individual still frames will seem to move smoothly; our brain will fill in the missing frames.

The rate of animation can be quite variable, but most people start to see 'jerkiness' if an animation is less than 15–20 fps (frames per second). Individual perception limits vary quite a bit from person to person. Cartoons and movies mostly rely on 24 fps, but there are a few people who see that as jerky. We will generally set a 20 or 30 fps rate on animation, even if the game is updating at 60 fps.

Like traditional cartoons, all computer graphics, even 3D, rely on there being enough frames to give the impression of the motion you want to represent over a sensible time for that motion to occur. You can make a character walk with just two images changing every 30 frames, but it looks better with four changing every 15, and better still with eight every seven or eight frames.

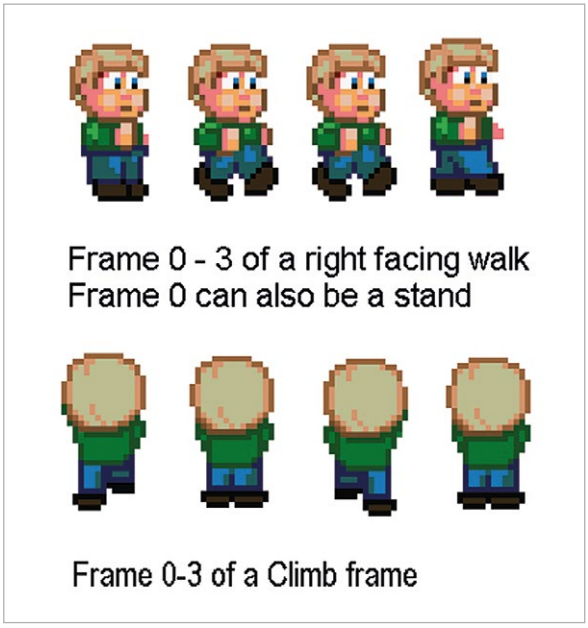


Figure 1 Simple individual frames

The more images for the sequence you have, the smoother the animation, but you are eating up your memory by storing so many individual sprite images. So a balance must be found.

Figure 1 shows two simple four-frame animations. Since we're only going to use a few frames for walking, we could cheat and reuse them for walking, standing, falling, and jumping, though at the moment we do need left- and right-facing. Each of these different actions we will call a 'state', but each state needs to keep track of what sequence of animation it is doing

KEEP ASSETS TIDY

At the moment we're not really doing that other than keeping them in a folder. We'll discuss this soon.

and how the control system will change depending on the state. For example, we can't really jump or climb if we are currently falling.

Separating the actions of our character into states means we can focus on the control that is relevant to that state and, where needed, signal a change in state – from walking to jumping, for example.

By defining each action as a state, we are making our first attempts at a state machine, where different code is called depending on the setting of that stage. That code can also change the state, allowing us to switch back and forward between states depending on whatever conditions we want to test. A good example might be when falling: we are only going to test to see if we should stop falling, and if we find ground under our feet, we switch into a standing frame, or perhaps even a duck-and-roll landing sequence state. Or a splatting into a bloody mess state if we are feeling a bit nasty and our hero fell a long way. Sadly, we don't really have enough graphics to let our imagination loose just yet.

Simple state management can be done with sequences of if-then-else conditions, but C/C++ gives us a much neater concept called a switch, which calls different code based on different 'case' statements. Refer to **Figure 2**.

Generally speaking, being able to keep the code for a specific state lets us focus on only dealing with the conditions that relate to that state. For example, when walking, we do not need to worry about jumping code, but we do need to test if we should be falling down. Our code becomes more about managing the state we are in at given points than trying to work out which of the five or six different abilities we give our character. This allows for expansion of ability and isolation of the code to provide those abilities.

```
typedef enum { // we define the names of our states like this
        STANDING = 0,
        WALKING,
        JUMPING,
        CLIMBING,
        FALLING,
        DROWNING
} StateVaues;

void Bob::BobsLogic()
{

        switch (CurrentState) // the Switch asks what is our current state represented as an enumerated value or int
        {
        case    STANDING:
                //Do code for a standing character
                break; // this ends the code
        case    WALKING:
                break; // notice we must end each section with break...if we don't it falls through
        case    JUMPING:
                //Do code for a jumping character
                break;  // this ends the code
        case    CLIMBING:
                //Do code for a climbing character
                break;  // this ends the code
        case    FALLING :
                //Do code for a falling character
                break;  // this ends the code

        case    DROWNING :
                //Do code for a drowingcharacter
                break;  // this ends the code

        default : // this is a special case in case we have not found code for a particular state value
                //do default code, which usually means we have forgotton something
                break;
        } // close the switch


}
```

Figure 2 How a switch keeps states separated

We'll start with a single screen map for now, which is called **Map2** in the code. As you can see in **Figure 3** over the page, this is a fairly simple platform game layout, but drawn the exact same way as our bat/ball map, which does create some issues we will discuss soon.

Remember that our characters do not interact directly with the screen: our screen only gives us a visual interpretation of what happens during the map interaction. So as long as we keep our code focused on the characters' interaction with the map, we can move our screen around any way we want.

Language
>C++

NAME:
MagPiHelloTriangle.cpp
SimpleObj.h/cpp
Game.h/cpp
OGL.h/cpp
Ball.h/cpp
NewBall.h/cpp
Paddle.h/cpp
Input.h/cpp

DOWNLOAD:
magpi.cc/YUdxXy

Using maps

Our bat-and-ball game touched on the concept of maps, since our screen was basically made up of an array which we interacted with. That array was a map, where each cell related to a specific tile on screen. But platform games take the concept of a map to a new level. Consider how many levels you can have in a game like Super Mario Bros; all these levels use the same quite limited and basic tile set, yet clearly there are thousands of cells making up each map. A cell in a map, however, can be represented by a single byte (though more typically a 32-bit integer) and allow us to have arrays of any size to represent quite huge numbers of levels.

What's more, those levels can be larger than the screen size, and by simply moving a screen view area – under the control of the principal character in our game – we can create scrolling. The view screen area moves as we want it to, but generally under the control of our principal character who drags the view along. We'll tackle that next time.

Our main character

Our new character is called Bob. The **Bob** class inherits from our simple objects and, like our previous balls, has some information on graphics and position which lets us draw him. Take a moment to review the code and the comments that explain what it's doing.

Bob's class is a little like our old **Bat** class: Bob is the only one who cares about the keys, so he's going to read the keys, move around, jump, and test if he's standing on something; if not, he's going to obey the laws of gravity and fall, though we won't splat him if he falls too far.

Gravity in game worlds acts very similarly to gravity in the real world: it's always there and pulling you down, but as long as there's ground beneath us we don't fall through. So we need to test if there's ground under our feet during our normal, non-gravity-defying states.

Gravity is usually represented as a speed in platform games: 0 when on the ground and whatever amount we find works best when not on the ground. To jump, we

KEEP CODE READABLE

Comments are great, but readable, self-descriptive code is better.

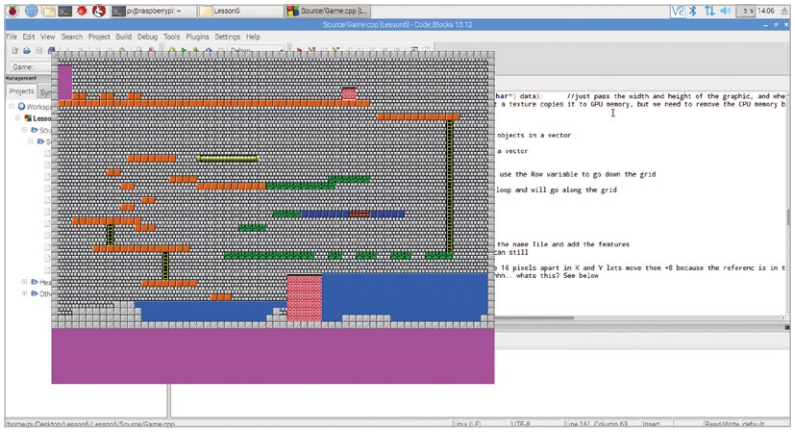


Figure 3
A nice new map!

simply give ourselves an upward speed and when gravity is in effect, we deduct it from the character's current speed, creating a reduction in his upward motion until it eventually goes negative, causing him to fall.

By adding a **Speed** value to the co-ordinates of our character, we can get much smoother – and, later, timed – movement in our characters. We can also use this speed concept for left and right motion, using friction to reduce it. The source code for Bob's movement shows this principle for both vertical and horizontal movement.

What am I standing on?

Testing the tile relies on us knowing the reference point of Bob. Usually that's the centre of the sprite, so a simple offset of half his height will get our testing point to be at his feet. In fact, we should add an extra one pixel, since we want to test what's just below his feet. Also, since our sprite is wider than a tile, we should really test a few points to make sure that if any part of the main graphics that make up the sprite are on a ground tile, we prevent the fall.

Now we have gravity, jumping is a simple case of giving him an upward speed that initially overcomes the force of gravity, but which is constantly reduced by gravity. With the result that what goes up, must come down, until it lands on a ground tile.

Jumping and climbing both start by pressing the Up key, so clearly a decision needs to be made when Up is pressed to determine if we climb or jump, and that will depend on our test of whether we are on a ladder or not. In order to climb a ladder, we must know which tile represents the ladder, in this case tile number 14.

Now, that is fine – we can test for our offset in the array to see if it has hit that tile number and if the correct up/down keys are pressed when were are over or above a ladder, we can change state to climb – but what if we have more than one ladder type? (We don't. But… what if we did?)

That's a lot of ifs

Our map is fairly simple, but we're already considering testing for two or three different concepts of what makes a ground tile we can stand on. At the moment, tile numbers 1, 2, 3, 4, 7, 8, 9, 10, and 11 appear to be types of ground. Our ladder, 14, is also in some ways a ground tile – so is a brick tile, so is a girder tile, so is a grass tile, etc. So we need to consider if sequentially testing tiles with a whole bunch of **`if`** statements is a wise move.

Well no, of course it's not, for all sorts of reasons. It is hard to manage when we add new tiles and it also makes the code test for the last tile in the list slower than the code test for the first.

A better idea is to consider that each tile has certain attributes, which we can encode into bit fields. Bits are the basic binary (on/off) components that make up all our data values, and it's possible to set and test individual bits (**Figure 4**). We don't need to be able to count in binary to use it; we just need to know that the bits are usable and very useful. In an **`int`** we have 32 bits, so we can store 32 on/off attributes.

Our simple game only really needs two concepts at the moment – Ground and Ladder – but we could later consider Metal and Brick as an attribute and trigger different types of walking sound effect.

Because we can use bit fields to define an attribute, we can in theory have 32 different attributes assigned to every tile, and that means that tiles which serve the same purpose but have different tile numbers can be treated the same in code.

The main difference from our viewpoint is that instead of testing for a tile number, we use that tile number to look up the attributes for that tile and test the attribute.

Next time

This is all so far so good, but clearly there's a massive issue: the speed of our game has fallen to unplayable levels! Do we know why? The source code has a **`#define`** called **`FastUpdate`** – try setting it to true and see what a difference it makes.

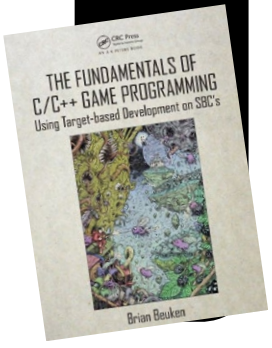
Next lesson we'll examine how to do scrolling, add some baddies, and also make more improvements to our updating to explain and further improve the performance of the tile draws. Sadly, there wasn't space this month to discuss text/font systems, so we'll try to fit that in next time to make score display and menu/instruction screens much simpler.

STATE MANAGEMENT MAKES LIFE EASY

Keep your states well defined and separate, and focus on making each state work before starting another.

LEARN MORE ABOUT C

Brian has a whole book on the subject of making games C and C++, called *The Fundamentals of C/C++ Game Programming: Using Target-based Development on SBCs*. Grab it here: **magpi.cc/nUkjEt**



How many ground tiles do we have, and how can we test?

Ground = 1
Ladder = 2

in Binary bit 0 = Ground
in Binary bit 1 = Ladder

Tile 1 has attribute 00000000000000000000000000000001
Tile 2,3,4,7,8,9,10,11, 15 are the same

Tile 14 has attribute 00000000000000000000000000000011
Because its both ground and ladder!

Figure 4 Encoding tile attributes into bit fields

HackSpace

TECHNOLOGY IN YOUR HANDS

THE NEW MAGAZINE FOR THE MODERN MAKER

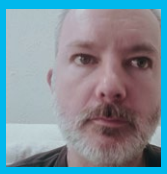
ISSUE #08

OUT NOW

hsmag.cc

Available on the App Store

GET IT ON Google Play



JODY CARTER

Jody is a primary school teacher and computing leader for a MAT. He is grateful for the Raspberry Pi camera as it saves him from having to stand outside in the rain to take bad pictures of wildlife.
twitter.com/codeyjody

BUILD A WILDLIFE CAMERA TRAP WITH OBJECT RECOGNITION

Uncover the goings-on in your garden, pond, or school playground when no one's looking with this easy-to-use Raspberry Pi camera trap

You'll Need

- Pi Camera Module **magpi.cc/camera**
- Pi NoIR Camera Module (optional) **magpi.cc/ircamera**
- ZeroCam NightVision (optional) **magpi.cc/qqXcLK**
- Waterproof container (like a jam jar)
- Blu Tack, Sugru, elastic bands, carabiners
- ZeroView (optional) **magpi.cc/2e8ghWt**

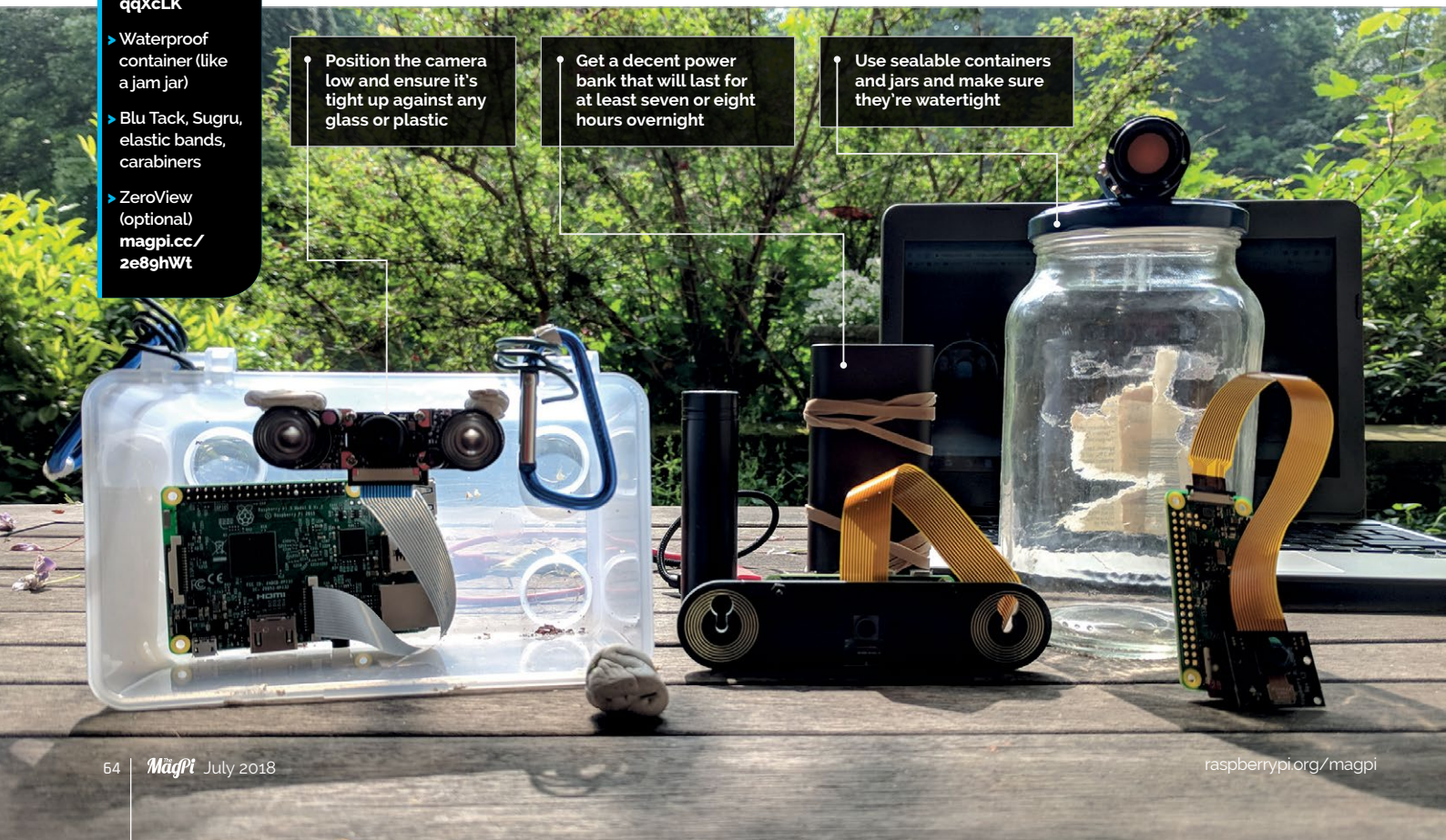
Ever wondered what lurks at the bottom of your garden at night, or which furry friends are visiting the school playground once all the children have gone home?

Using a Raspberry Pi and camera, along with Google's Vision API, is a cheap but effective way to capture some excellent close-ups of foxes, birds, mice, squirrels and badgers, and to tweet the results.

Using Google's Vision API makes it really easy to get AI to classify our own images. We'll install and set up some motion detection, link to our Vision API, and then tweet the picture if there's a bird in it. It's assumed you are using a new Raspbian installation on your Raspberry Pi and you have your Pi camera set up (whichever model you're using). You will also need a Twitter account and a Google account to set up the APIs.

Motion detection with Pi-timolo

There are many different motion-detection libraries available, but Pi-timolo was chosen as it is easy to edit the Python source code. To install, open a Terminal window and enter:


Position the camera low and ensure it's tight up against any glass or plastic
Get a decent power bank that will last for at least seven or eight hours overnight
Use sealable containers and jars and make sure they're watertight

Above Unleash your inner *Springwatch*

```
cd ~
wget https://raw.github.com/pageauc/pi-timolo/master/source/pi-timolo-install.sh
chmod +x pi-timolo-install.sh
./pi-timolo-install.sh
```

Once installed, test it by typing in **cd ~/pi-timolo** and then **./pi-timolo.py** to run the Python script. At this point, you should be alerted to any errors such as the camera not being installed correctly, otherwise the script will run and you should see debug info in the Terminal window. Check the pictures by waving your hand in front of the camera, then looking in Pi-timolo > Media Recent > Motion. You may need to change the image size and orientation of the camera; in the Terminal window, enter **nano config.py** and edit these variables: **imageWidth**, **imageHeight**, and **imageRotation**.

While we're here, if you get a lot of false positives, try changing the **motionTrackMinArea** and **motionTrackTrigLen** variables and experiment with the values by increasing to reduce sensitivity. See the Pi-timolo GitHub repo (**magpi.cc/PFqFSJ**) for more details.

There's also going to be some editing of the **pi-timolo.py** file, so don't close the Terminal window. Code needs to be added to import some Python libraries (**Listing 1**), and also added to the function **userMotionCodeHere()** to check with the Vision

Listing 1

```
# add this in at the very top, under print('Loading ....') along with the other libraries imported
import io
import tweepy
from google.cloud import vision
from google.cloud.vision import types
from google.cloud import storage
```

Listing 2

Language
>PYTHON 2

NAME:
motion_detection_code.py

DOWNLOAD:
magpi.cc/qoRuSW

```
# search for userMotionCodeHere.
# There will be 2 results,
# edit the second so you are passing filename to the function
userMotionCodeHere(filename)

# make sure you include filename as a parameter in the function
def userMotionCodeHere(filename):
    # we need to create an instance of the Google Vision API
    client = storage.Client()
    # instantiates a client
    client = vision.ImageAnnotatorClient()

    # loads the image into memory
    with io.open(filename, 'rb') as image_file:
        content = image_file.read()

    image = types.Image(content=content)

    # performs label detection on the image file
    response = client.label_detection(image=image)
    # pass the response into a variable
    labels = response.label_annotations

    # we have our labels, now create a string to add to the text
    # for debugging - let's see what Google thinks is in the image
    print('Labels:')
    # add labels to our tweet text
    tweetText = "Labels: "
    animalInPic = False
    for label in labels:
        print(label.description)
        tweetText = tweetText + " " + label.description
        # edit this line to change the animal you want to detect
        if "bird" in tweetText: animalInPic = true

    # set up Tweepy
    # consumer keys and access tokens, used for authorisation
    consumer_key = 'XXX'
    consumer_secret = 'XXX'
    access_token = 'XXX'
    access_token_secret = 'XXX'

    # authorisation process, using the keys and tokens
    auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
    auth.set_access_token(access_token, access_token_secret)

    # creation of the actual interface, using authentication
    api = tweepy.API(auth)

    # send the tweet with photo and message
    photo_path = filename
    # only send tweet if it contains a desired animal
    if animalInPic:
        api.update_with_media(photo_path, status=tweetText)

    return
```

Above Get great photos with night-vision cameras

API before tweeting (**Listing 2**). We can do this now in preparation of setting up our Google and Twitter API. You should still be in the **Pi-timolo** folder, so type **`nano pi-timolo.py`** and add the imports at the top of the file. Next, press **CTRL+W** to use the search option to find the **`UserMotionCodeHere()`** function and where

> We will be using Google Label Detection, which returns a list it associates with the image

it's called from. Add the new code into the function (line 240), before the **`return`** line. Also locate where the function is being called from (line 1798), to pass the image file name and path. Press **CTRL+X** then **Y** and **RETURN** to save. Next, we'll set up the APIs.

Animal detection and tweeting

We will be using Google Label Detection, which returns a list it associates with the image. First off, you will need to install the Google Cloud Vision libraries on your Raspberry Pi, so type **`pip install --upgrade google-cloud-vision`** into your Terminal window. Once finished, run **`pip install google-cloud-storage`**.

Now you need authorisation, by going to **magpi.cc/xVLSVa** to set up an account. Click on the Manage Resources link and create a new project (you may need to log in or create a Google account). Go to the API Dashboard and search for and enable the Vision PI. Then go to API & Services > Credentials, click on Create Credentials > Service Account Key > New Service Account from the drop-down. Don't worry about choosing a Role. Click Create and you'll be prompted to download a JSON file. You need this as it contains your service account key to allow you to make calls to the API locally. Rename and move the JSON file into your **Pi-timolo** folder and make a note of the file path. Next, go back to **pi-timolo.py** and add the line: **`os.environ["GOOGLE_APPLICATION_CREDENTIALS"] = "path_to_your_.json_credential_file"`** below **`import os`** to reference the credentials in your JSON file.

Finally, set up a Twitter account if you haven't already and install Tweepy by entering **`sudo pip install tweepy`** into a Terminal window. Once set up, visit **apps.twitter.com** and create a new app, then click on Keys and Access Tokens. Edit the code in **`userMotionCodeHere()`** with your own consumer and access info, labelled as 'XXX' in the code listing. Finally, place your camera in front of your bird feeder and run **`./pi-timolo.py`**. Any pictures taken of a bird should now be tweeted! If you want to identify a different animal, change the line **`if "bird" in tweetText: animalInPic = true`**.

Please note that although the API works well, it can't always discern exactly what's in the picture, especially if partially in view. It also won't distinguish between types of bird, but you should have more success with mammals. You can test the API out with some of your pictures at **magpi.cc/EBzDam** and visit **twitter.com/pibirdbrain** to see example tweets. Good luck and happy tweeting!

CoderDojo

Can I start a CoderDojo club in my local area?

CoderDojo is a global network of free, volunteer-led, project-based programming clubs for children aged 7–17. Dojos are championed by individuals all around the world who are passionate about giving young people the opportunity to learn to code.

Starting a Dojo is a fun and incredibly rewarding experience

You don't need to possess technical skills to start a Dojo. The most important attribute is that you can bring people together for a shared goal.

We're ready to support you by providing:

- **Learning resources and guides**
- **A free event management system**
- **Certificate templates, digital badges, and giveaways**

"I started a Dojo to give my kids a place to meet other children also interested in programming and making games. I get to see them making new friends, learning from one other, and they loved it. Realising how I had created such a wonderful place for children has ignited a spark in me."

- Maroes, CoderDojo NL

Start your own club. Join us at CoderDojo.com

The CoderDojo Foundation is Irish Registered Charity (524255) CHY20812
and is part of the Raspberry Pi Foundation, UK registered charity 1129409

Part of Raspberry Pi

FREQUENTLY ASKED QUESTIONS

NEED A PROBLEM SOLVED?

Email **magpi@raspberrypi.org** or find us on **raspberrypi.org/forums** to feature in a future issue.

Your technical hardware and software problems solved...

LEARNING RESOURCES

DOES RASPBERRY PI PROVIDE PROJECT EXAMPLES AND HELP?

Official projects site

Part of the Raspberry Pi Foundation's mission is to make sure there are plenty of free learning resources for students and educators to make use of, and you can find a huge range of guides on **projects.raspberrypi.org**. Many have been translated into a variety of languages as well!

Raspberry Pi Forum

The forum is the number one place to go if you're having trouble with a project. There's a wealth of information there which you can find with the search bar, and an active community that will try to help you with your problem: **raspberrypi.org/forums**.

Raspberry Pi Press

The MagPi is published by the Raspberry Pi Press (**store.rpipress.cc**), and everything made there is available for free as a PDF. While it's not all specifically aimed towards education like our *Beginner's Book*, there are a lot of guides and inspirational projects that will help you on your Raspberry Pi journey.

DOES RASPBERRY PI PROVIDE RESOURCES FOR TEACHERS?

Picademy

The Raspberry Pi Foundation provides free teacher training via the Picademy scheme, currently in the UK and USA. At these two-day events, teachers and educators can learn everything they need to know about using the Raspberry Pi and using it to teach kids about computing and digital making. Find out more here: **magpi.cc/2Bakf96**.

After-school clubs

Code Club (**codeclub.org.uk**) and CoderDojo (**coderdojo.com**) are networks of weekend and after-school clubs that are part of the Raspberry Pi Foundation. They offer free resources for kids to learn about computing, which are used around the world.

Hello World

The magazine for educators is available as a free PDF and you can also get a print subscription of it for free. It includes lesson plans and features aimed towards teachers and other educators who are working in computer science fields: **helloworld.cc**.

THE MAGAZINE FOR COMPUTING & DIGITAL MAKING EDUCATORS
(Hello World)
DATABASES AND BIG DATA
INSPIRING ENGINEERS!
Coding and INCLUSION
ORGANISING A COMPUTING DAY OUT
PSSST! IS IT OKAY TO COPY?

ARE THERE ANY UNOFFICIAL RESOURCES YOU RECOMMEND?

Adafruit

These US-based makers specialise in components and add-ons for the Raspberry Pi, as well as awesome tutorials on how to make cool things. This includes handheld retro consoles, face-recognition locks, and so much more! Check them out here: **adafruit.com**.

Hackster.io

You can always find people sharing great Raspberry Pi tutorials on **hackster.io** (that's the web address) from all around the world! Not all of them may be to your taste, but the team also do regular highlights of the best projects which you can use for inspiration.

element14

Farnell's community side is a great way to see people's building process, thanks to various competitions revolving around different challenges. You can even enter a challenge yourself to try to win something. Check it out here: **magpi.cc/GrmbXe**.

FROM THE RASPBERRY PI FAQ RASPBERRYPI.ORG/HELP

WHAT ARE THE DIMENSIONS OF THE RASPBERRY PI?

The Raspberry Pi Model B versions measure 85.60 × 56 × 21 mm (roughly 3.37 × 2.21 × 0.83 inches), with a little overlap for the microSD card and connectors which project over the edges. They weigh 45 g. The Pi Zero and Pi Zero W measure 65 × 30 × 5.4 mm (roughly 2.56 × 1.18 × 0.20 inches) and weigh 9 g. For the mechanical outline, please see the documentation: **magpi.cc/xhPBSq**.

WHAT HARDWARE DOCUMENTATION IS AVAILABLE?

All available documentation is in our documentation repository: **magpi.cc/ujsAel**.

WHAT SOC ARE YOU USING?

All models of Raspberry Pi use Broadcom SoCs, containing a VideoCore IV GPU but with various ARM CPU cores.

The original Raspberry Pi used a Broadcom BCM2835. This contains a single-core ARM1176JZFS with floating point, running at 700MHz, and a VideoCore IV GPU. The GPU is capable of Blu-ray-quality playback, using H.264 at 40MBps. It has a fast 3D core, accessed using the supplied OpenGL ES 2.0 and OpenVG libraries.

The Pi 2 Model B originally used the Broadcom BCM2836. This contains a quad-core ARM Cortex-A7 processor with floating point and NEON, running at 900MHz, and the same VideoCore IV GPU that is in the earlier models of Raspberry Pi.

The Pi 3 Model B uses the Broadcom BCM2837, containing a quad-core ARM Cortex-A53 running at 1.2GHz. Its GPU capabilities are equivalent to the Pi 2. Newer Pi 2 boards now use the same SoC as the Pi 3, but downclocked to match the speed of the original Pi 2 SoC.

The new Pi 3 Model B+ uses the Broadcom BCM2837B0, containing a quad-core ARM Cortex-A53 running at 1.4 GHz.

WHAT IS A SOC?

A system-on-a-chip (SoC) is a method of placing all necessary electronics for running a computer on a single chip. Instead of having an individual chip for the CPU, GPU, USB controller, RAM, etc., everything is compressed down into one tidy package.

WHY DID YOU SELECT THE SOC?

Cost and performance.

HOW DOES IT BOOT?

As standard, all the files necessary for booting are installed in a FAT32 partition of the microSD card. The latest firmware/software, however, allows booting to be set up without a microSD card, for example from a USB stick or other mass storage device.

THE MAGPI APP

Having trouble with The MagPi on the App Store or Google Play? Here are your most common questions answered.

How do I find The MagPi on Google Play or the App Store?
All you have to do is go to the search bar and type 'The MagPi' or 'Raspberry Pi' to find us.

I've subscribed to the digital edition and I can't sign in to restore my purchases. Please help!
Since your The MagPi purchases are linked to your Google or Apple accounts, there's no need to sign in at all. If you'd like to re-download your purchases on your current device, or make your purchases available on other devices, all you need to do is hit Menu on the home screen, then Restore Purchases on the sidebar.

How can I search the digital magazine for keywords?
Finding direct references is really easy with The MagPi app – all you have to do is tap the screen to get the app's GUI to show, and then press the small magnifying glass icon in the top-right corner of the screen. Now, just type in your search term to find the relevant results.

BUY IN PRINT WORLDWIDE! magpi.cc/store
The MagPi
RUN ANDROID ON RASPBERRY PI
Use Android Apps
Build Touchscreen Devices
Control Electronics
MAKE YOUR OWN OSCILLOSCOPE
MAGIC: THE GATHERING CARD COUNTER
BUILD A WILDLIFE CAMERA
HERE COMES THE SUN
ADD AI TO YOUR PROJECTS WITH TENSORFLOW

GET IT ON Google Play

Available on the App Store

RICHARD HAYLER

Richard runs the Raspberry Pi Citizen Science programme, including the Weather Station project. He's an engineer who's easily distracted by comics, music, rugby, and LEGO. His quest to combine them into a single activity is ongoing.
magpi.cc/EBrOjB

BUILD A WEATHER STATION

Hardware You'll Need

- Raspberry Pi with wireless connectivity (or a wireless dongle)
- BME280 pressure, temperature, and humidity sensor **magpi.cc/lFGBDk**
- DS18B20 digital thermal probe (with 1m lead)
- 2 × 4.7kΩ resistors
- Some 5mm-pitch PCB mount screw terminal blocks
- A breadboard, some jumper wires
- Anemometer, wind vane, and rain gauge **magpi.cc/txkqdr**
- 2 × RJ11 breakout boards (optional)
- MCP3008 analogue-to-digital converter integrated circuit **magpi.cc/BlHlKm**
- Weatherproof enclosures; recommended products are the IP55 Thermoplastic 7 Entry Junction Box Enclosure (**magpi.cc/myRZHW**) and IP55 Thermoplastic 10 Entry Junction Box Enclosure (**magpi.cc/uYvzUZ**)

BUILD YOUR WEATHER STATION TO MEASURE TEMPERATURE, HUMIDITY, AND ATMOSPHERIC PRESSURE, WIND AND RAINFALL. COME RAIN OR SHINE YOU'LL BE ABLE TO SEE WHAT'S COMING

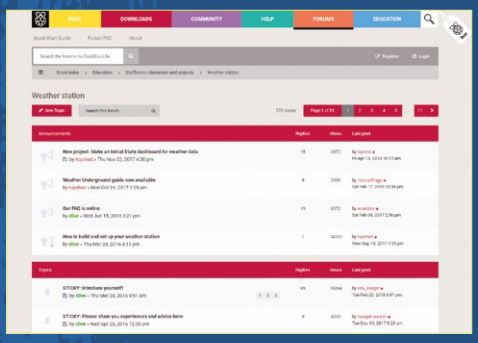
In 2016, Raspberry Pi sent out nearly 1000 Oracle Raspberry Pi Weather Stations to schools from around the world who had applied to receive these kits.

The Weather Station kit enables a Raspberry Pi to collect weather data using a variety of sensors. These kits have been very popular. If you're one of the many people who has been wanting to get hold of one, this guide will take you through the process of building your own.

This is an advanced project, both in terms of electronics and programming. You can find the full project, with even more details, on the Raspberry Pi website: **magpi.cc/SYWwma**.

FORUM HELP

If you get stuck while making this project, you can find help on the Raspberry Pi forums. There is a dedicated board for the Weather Station.
magpi.cc/YPcmCa



SOFTWARE DOWNLOAD

You'll need the Oracle Raspberry Pi Weather Station software. You don't need to install it, but you'll use some of the Python programs. Clone the GitHub repository by opening a Terminal window and typing:

```
git clone https://github.com/RaspberryPiFoundation/weather-station
```

Now install the BME280 Python library:

```
sudo pip3 install RPi.bme280
```

You can find more information on this library at the Python Package Index (**magpi.cc/dhSyei**).

And install the MariaDB database server software:

```
sudo apt-get install -y mariadb-server mariadb-client libmariadbclient-dev

sudo pip3 install mysqlclient
```

The wind vane detects the direction the wind is coming from
The anemometer has three arms with scoops on the end that catch the wind and cause the arms to spin. This measures the wind speed
A 3D-printable mount is used to secure the Raspberry Pi inside the larger box
The smaller box contains a BME280 integrated environmental sensor. This provides data on the humidity, pressure, and temperature
RASPBERRY PI
WEATHER STATION

Measure Temperature and Humidity

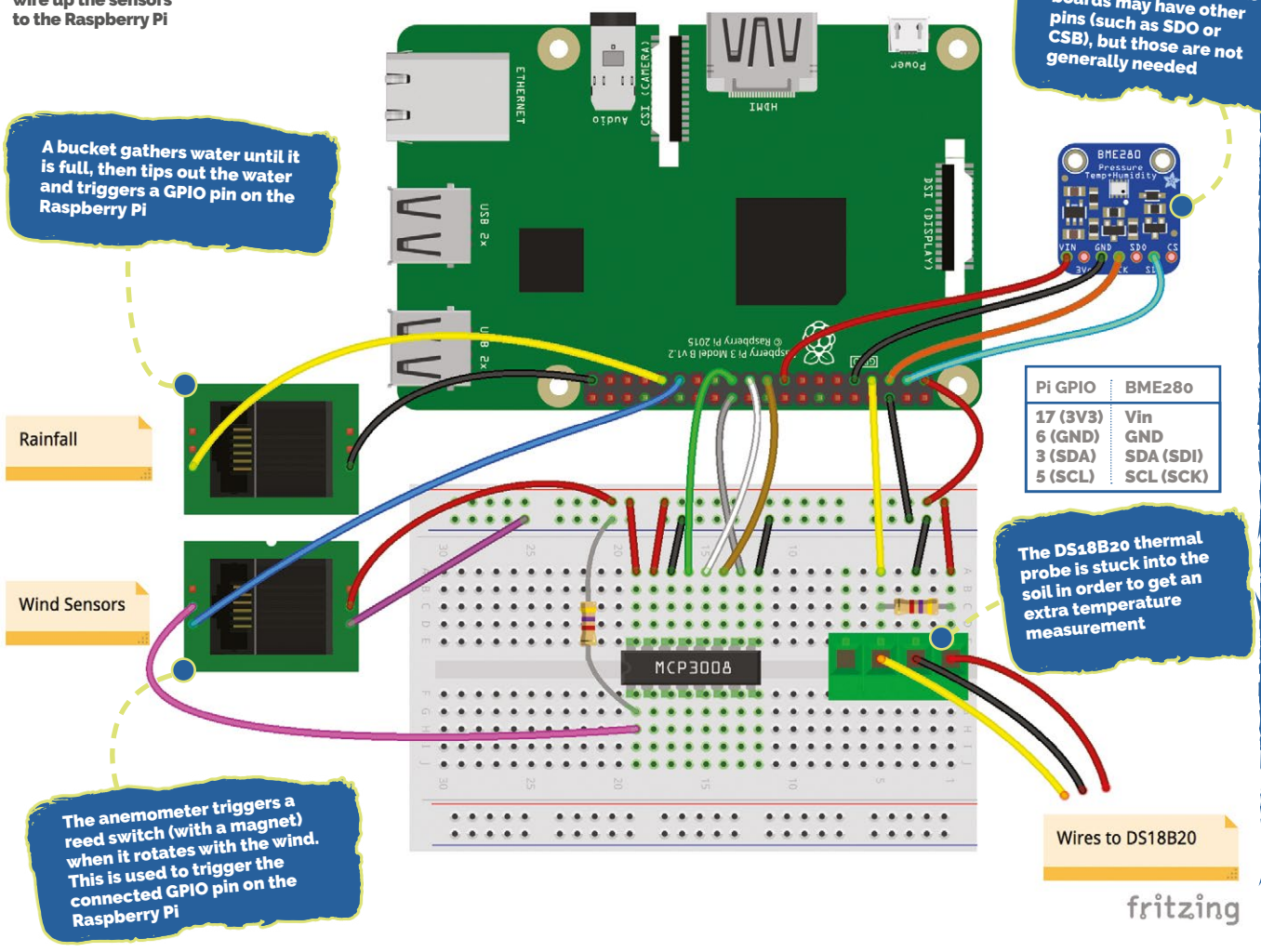
TURN YOUR RASPBERRY PI INTO A HEAT AND HUMIDITY MACHINE USING A BME280 SENSOR

The BME280 sensor is a digital sensor that can measure temperature, humidity, and atmospheric pressure.

It is available in a number of breakout boards from popular manufacturers such as Adafruit and SparkFun. This guide assumes you have the Adafruit package (**magpi.cc/lFGBDk**), but the instructions should be applicable to most versions.

One thing to check is that the I²C address is correct: for the Adafruit models it is 0x77 (as shown in the **bme280_sensor.py** code), but other versions can have different addresses (0x76 is a common alternative, as shown in **bme280_sensor_2.py**).

Figure 1 How to wire up the sensors to the Raspberry Pi


Wire up your BME280 as shown in the table below. The BME280 is connected to the GPIO pins on the Raspberry Pi. Some other breakout boards may have other pins (such as SDO or CSB), but those are not generally needed
A bucket gathers water until it is full, then tips out the water and triggers a GPIO pin on the Raspberry Pi
Rainfall
Wind Sensors
The anemometer triggers a reed switch (with a magnet) when it rotates with the wind. This is used to trigger the connected GPIO pin on the Raspberry Pi
The DS18B20 thermal probe is stuck into the soil in order to get an extra temperature measurement
Wires to DS18B20
MCP3008
BME280
Pressure
Temp+Humidity
VIN GND SDO CS
ETHERNET
Audio
CSI (CAMERA)
HDMI
Power
USB 2x
USB 2x
DSI (DISPLAY)
Raspberry Pi 3 Model B V1.2
© Raspberry Pi 2015
fritzing

Pi GPIO	BME280
17 (3V3)	Vin
6 (GND)	GND
3 (SDA)	SDA (SDI)
5 (SCL)	SCL (SCK)

STEP 1

OPEN IDLE

In Raspbian, select Menu > Programming > Python 3 (IDLE). Create a new Python file, save it as **bme280_sensor.py** in the **/home/pi/weather-station** directory and add the code from **bme_280_sensor.py** to it.

STEP 2

DEEP BREATH

While the code is running, exhale onto the BME280 sensor. You should see the humidity value (and possibly the temperature) increase. Press **CTRL+C** to exit the program.

STEP 3

UPDATE THE CODE

Replace the `while True` loop with a function called `read_all()` that returns the humidity, pressure, and temperature readings, in that order. Update your code using **bme_280_sensor_2.py** as a guide.

STEP 4

GROUND TEMPERATURE

The BME280 will report the air temperature, but this can be warmer than the ground, particularly if it's frosty. A DS18B20 thermal probe (**magpi.cc/JILuTd**) stuck into the soil is a useful supplemental measurement.

STEP 5

WIRING THE SENSOR

Normally the DS18B20 comes with three bare wires, so the easiest way to test it is to use PCB mount screw terminal blocks that can be plugged into breadboards. Add your DS18B20 to your circuit as shown in **Figure 1**.

STEP 6

EDIT THE BOOT

Open the file **/boot/config.txt:**

```
sudo nano /boot/config.txt
```

Edit it by adding the line below at the bottom:

```
dtoverlay=w1-gpio
```

Then open **/etc/modules**:

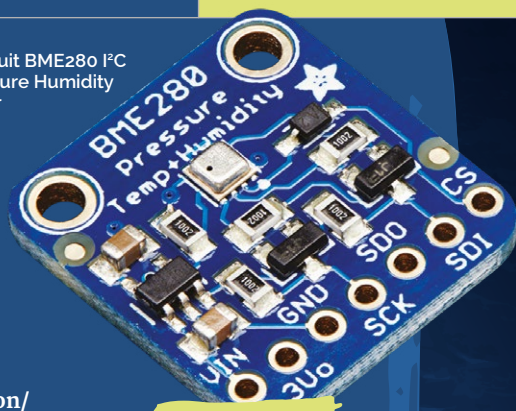
```
sudo nano /etc/modules
```

Add the lines below at the bottom of the file:

```
w1-gpio
w1-therm
```

Reboot the Raspberry Pi.

Open the file **/home/pi/weather-station/ds18b20_therm.py** in IDLE and run it (see 'Software download' box on page 71 to use Git to clone this file). You should see the temperature printed out in the Python Shell window.



Right The Adafruit BME280 I²C or SPI Temperature Humidity Pressure Sensor

CODE

Language: Python
Name: bme280_sensor.py, bme280_sensor_2.py
Link: magpi.cc/GhJlde
Name: ds18b20_therm.py
Link: magpi.cc/NQGLtb

bme280_sensor.py

```
01. import bme280
02. import smbus2
03. from time import sleep
04.
05. port = 1
06. address = 0x77 # Adafruit BME280 address. Other BME280s
    may be different
07. bus = smbus2.SMBus(port)
08.
09. bme280.load_calibration_params(bus,address)
10.
11. while True:
12.     bme280_data = bme280.sample(bus,address)
13.     humidity  = bme280_data.humidity
14.     pressure  = bme280_data.pressure
15.     ambient_temperature = bme280_data.temperature
16.     print(humidity, pressure, ambient_temperature)
17.     sleep(1)
```

bme280_sensor_2.py

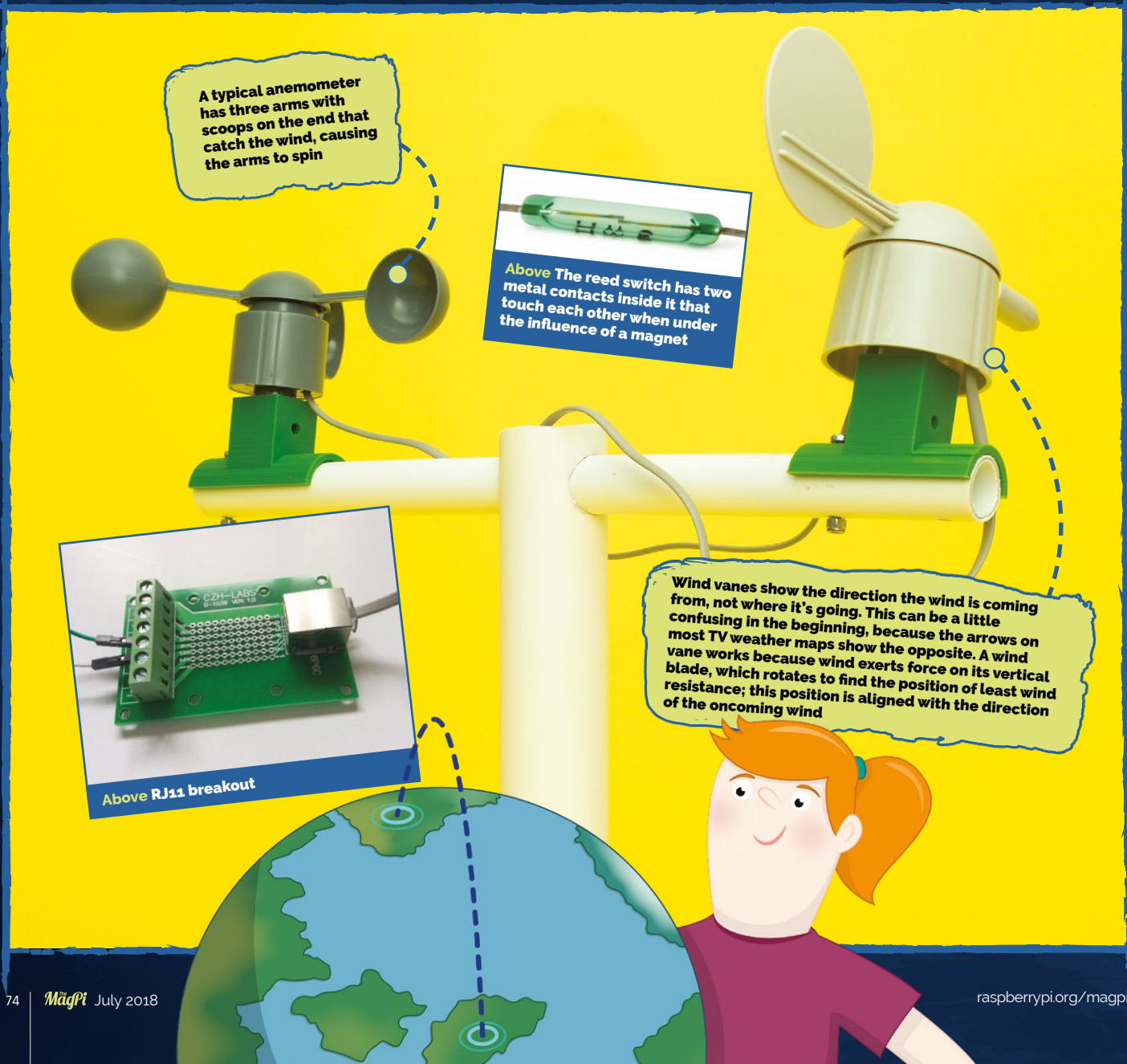
```
01. import bme280
02. import smbus2
03. from time import sleep
04.
05. port = 1
06. address = 0x76
07. bus = smbus2.SMBus(port)
08.
09. bme280_load_calibration_params(bus, address)
10.
11. def read_all():
12.     bme280_data = bme.sample(bus,address)
13.     return bme280_data.humidity, bme280_data.pressure,
    bme280_data.temperature
```

Air Quality and Wind Speed

YOU'VE DEALT WITH HEAT, NOW IT'S TIME TO TACKLE ANOTHER ELEMENT: AIR

So far, all the sensors you've used have been passive electronic sensors that just sit there and make measurements of their surroundings. However, to measure things like rainfall and wind speed/direction, you'll need to use active mechanical devices that physically interact with the environment.

The original Oracle Weather Station kit employed popular wind and rain sensors (**magpi.cc/mkxjNE**) that are used in many consumer weather stations. These are the recommended sensors to use, as they are robust and reliable. Their data sheet (**magpi.cc/ibPeBX**) gives more information about the sensors' size and construction.


A typical anemometer has three arms with scoops on the end that catch the wind, causing the arms to spin
Above The reed switch has two metal contacts inside it that touch each other when under the influence of a magnet
Above RJ11 breakout
Wind vanes show the direction the wind is coming from, not where it's going. This can be a little confusing in the beginning, because the arrows on most TV weather maps show the opposite. A wind vane works because wind exerts force on its vertical blade, which rotates to find the position of least wind resistance; this position is aligned with the direction of the oncoming wind

STEP 1
CONNECTION

Anemometers normally have two wires. Connect one to a GND pin, the other to GPIO 05. If using the RJ11 connectors, the anemometer uses the middle two wires of the cable: normally pins 3 and 4 on RJ11 breakout boards.

STEP 2
TEST WITH CODE

Open IDLE, create a new Python file, and save it as **wind_speed.py** in the **/home/pi/weather-station** directory. Add the code from the **wind_speed.py** listing. Save and run your code. Test it by manually turning the arms of the anemometer.

STEP 3
WIND SPEED

The anemometer produces two signals per spin, so you can count the number of full rotations of the sensor by halving the number of detected inputs. This can then be used to calculate the wind speed:

speed = distance / time

To calculate speed, you need to know the distance travelled in a certain amount of time. Measuring time is fairly straightforward, and you can count the number of signals over the course of a fixed time period, for example 5 seconds. The distance travelled by one of the cups will be equal to the number of rotations multiplied by the distance around the edge of the circle (circumference):

speed = (rotations × circumference) / time

The circumference can be calculated as long as you know either the radius of the circle. You can discover the radius made by the anemometer by measuring the distance from the centre to the edge of one of the cups. Once you know the radius, you can find the circumference with the formula $2 \times \pi \times$ radius. Don't forget that a whole rotation generates two signals, so you'll need to halve the number of signals detected:

speed = ((signals/2) × (2 × π × radius)) / time

The radius for the recommended anemometers used by the original Oracle Weather Station is 9.0 cm, and that is the figure which is used in the code example. Don't forget to change this value if your anemometer has different dimensions!

To implement this formula in Python, you can use the `math` library. For example, if you measured 17 signals from your anemometer in 5 seconds, your wind speed could be calculated like the code in **wind_speed.py**.

wind_speed.py

CODE

Language: Python
Name: wind_speed.py
magpi.cc/ABckLC

```
01. from gpiozero import Button
02. import time
03. import math
04.
05. wind_count = 0  # Counts how
    many half-rotations
06. radius_cm = 9.0 # Radius of your anemometer
07. wind_interval = 5     # How often (secs) to report speed
08.
09. # Every half-rotation, add 1 to count
10. def spin():
11.     global wind_count
12.     wind_count = wind_count + 1
13.     # print("spin" + str(wind_count))
14.
15. # Calculate the wind speed
16. def calculate_speed(time_sec):
17.         global wind_count
18.         circumference_cm = (2 * math.pi) * radius_cm
19.         rotations = wind_count / 2.0
20.
21.         # Calculate distance travelled by a cup in cm
22.         dist_cm = circumference_cm * rotations
23.
24.         speed = dist_cm / time_sec
25.
26.         return speed
27.
28. wind_speed_sensor = Button(5)
29. wind_speed_sensor.when_pressed = spin
30.
31. # Loop to measure wind speed and report at 5-second intervals
32. while True:
33.         wind_count = 0
34.         time.sleep(wind_interval)
35.         print( calculate_speed(wind_interval), "cm/h")
```

STEP 4
WIND DIRECTION

If you look inside the recommended wind vane, you'll see eight reed switches arranged like the spokes of a wheel. These can be used to measure the wind direction. In order to do so, you'll need to be able to measure the resistance produced by the sensor and convert that into an angle value. There are several steps in this process, and you need an analogue-to-digital converter (ADC). A popular and versatile ADC is the MCP3008 (**magpi.cc/BlHlKm**)

Measuring the wind direction is a bonus project and you can read how to set up the rotating magnets and ADC here: **magpi.cc/nZNLga**.

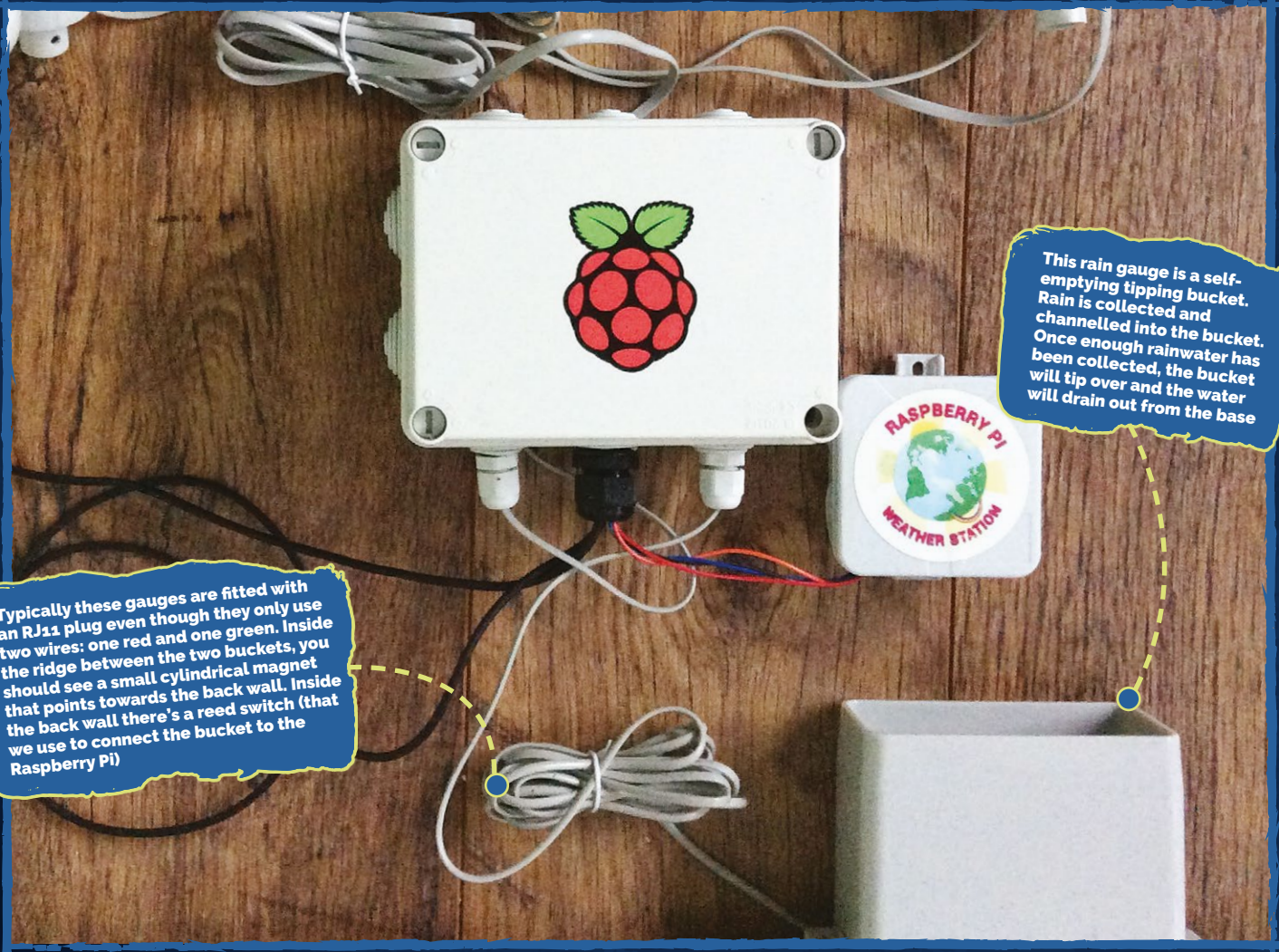
MEASURING RAINFALL

YOU CAN'T STOP IT RAINING, BUT YOU CAN MEASURE IT IN A BUCKET

Most rain gauges measure precipitation in millimetres in height collected on one square metre during a certain time period.

The recommended rain gauge sensor supplied with the Raspberry Pi Weather Station kit is actually a simple mechanical device included with the Argent Systems Wind / Rain Sensor (**magpi.cc/bWcYpV**).

This rain gauge is basically a self-emptying tipping bucket. Rain is collected and channelled into the bucket. Once enough rainwater has been collected, the bucket will tip over, the water will drain out from the base, and the opposite bucket will come up into position.


RASPBERRY PI
WEATHER STATION
This rain gauge is a self-emptying tipping bucket. Rain is collected and channelled into the bucket. Once enough rainwater has been collected, the bucket will tip over and the water will drain out from the base
Typically these gauges are fitted with an RJ11 plug even though they only use two wires: one red and one green. Inside the ridge between the two buckets, you should see a small cylindrical magnet that points towards the back wall. Inside the back wall there's a reed switch (that we use to connect the bucket to the Raspberry Pi)

STEP 1

TIP THE BUCKET

The product data sheet (**magpi.cc/ltofJB**) tells us that 0.2794 mm of rain will tip the bucket. You can multiply this by the number of tips to calculate the amount of rainfall. If you're using a different type of rain gauge, then you should consult the relevant data sheet or determine the volume of water required experimentally.

STEP 2

INSIDE THE BUCKET

The top of the back wall does come off if you want to see inside; just pull on the flat end gently and it should release. Inside there's a small circuit board that you can remove to examine. In the middle of it you will see the reed switch. Replace the circuit board and back wall lid before continuing.

When one of the buckets tips, the magnet passes the reed switch, causing it to close momentarily. So, just like with the anemometer, if you connect the rain gauge to a GPIO pin on the Raspberry Pi, you can treat it like a button and count the number of 'presses' to calculate rainfall.

STEP 3

WIRE IT UP

To test the rain gauge, you'll need to either remove the RJ11 connector and strip the wires, or make use of an RJ11 breakout board.

The Oracle Weather Station rain gauge is connected to GPIO 06, so for consistency, use the same pin for your device. (See the main **Figure 1** circuit diagram on page 72.)

STEP 4

WRITE THE CODE

Using the code you wrote for the anemometer as a starting point, write a program called **rainfall.py** (saved in the **/home/pi/weather-station** directory) to detect when the rain gauge bucket has tipped. It should print out a running count of how many bucket tips have occurred.

Now that you can count bucket tips, you need to convert this into a height of water that equals the amount of rain that has fallen.

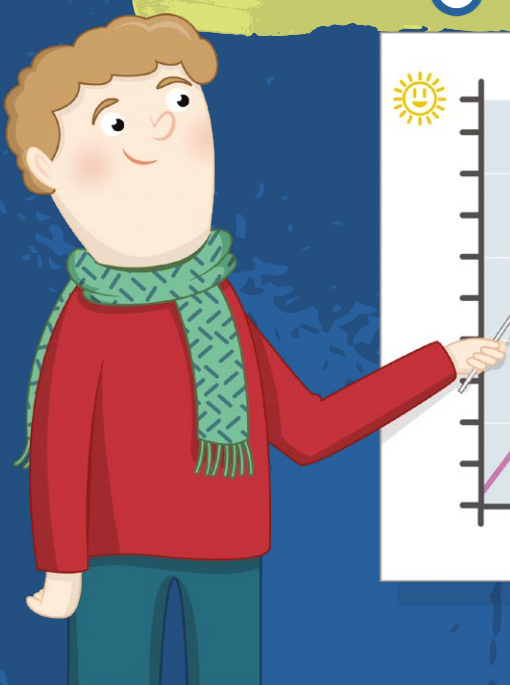
Finally, we use a function named `reset_rainfall` to reset the count of bucket tips so that it starts at 0 again. You will need this function for your fully operational weather station.

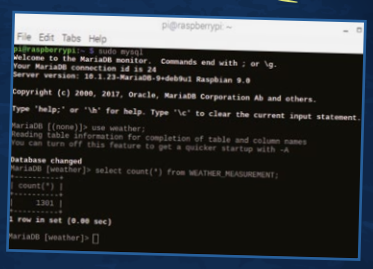
rainfall.py

```
01. from gpiozero import Button
02.
03. rain_sensor = Button(6)
04. BUCKET_SIZE = 0.2794
05. count = 0
06.
07. def bucket_tipped():
08.     global count
09.     count = count + 1
10.     print (count * BUCKET_SIZE)
11.
12. def reset_rainfall():
13.     global count
14.     count = 0
15.
16. rain_sensor.when_pressed =
    bucket_tipped
```

CODE

Language: Python
Name: rainfall.py
magpi.cc/uNgWwe





STORING MEASUREMENTS IN A DATABASE

One of the best ways to store your weather data is in a database. Databases can store very large numbers of records efficiently and make it easier to sort, search, and analyse your data.

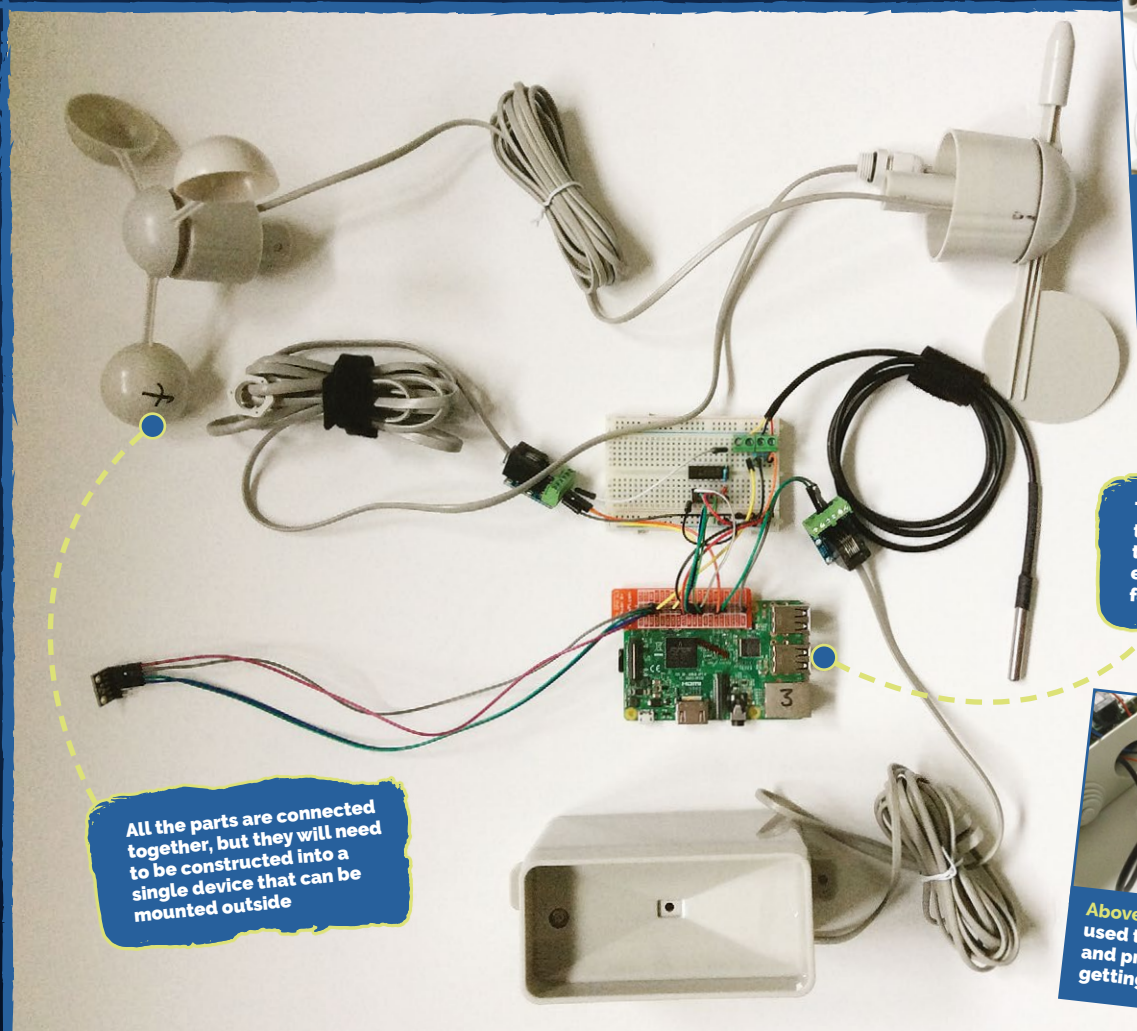
There are many different pieces of database software, and MariaDB is a good, versatile general-purpose product. You should have already installed it – if not, head back to the 'Software download' section on page 71 and install it.

Now follow the instructions at magpi.cc/ltofJB to learn how to set up and run your database.

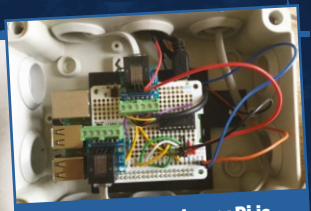
Assemble the Weather Station

NOW THAT YOU HAVE ALL YOUR COMPONENTS WORKING, IT'S TIME TO TURN THEM INTO A DEVICE THAT CAN LIVE IN THE GREAT OUTDOORS

So that your weather station can upload data to somewhere you can view and analyse it, it will need some form of internet connection. Using WiFi is typically the easiest way to do this, but you can use the Ethernet connection if that works better for your location. There's a guide about setting up wireless connectivity on your Raspberry Pi at **magpi.cc/fTclJO**, and some special hints for getting wireless access working with a weather station are at **magpi.cc/klShuT**.



All the parts are connected together, but they will need to be constructed into a single device that can be mounted outside



Above The Raspberry Pi is fitted in a large box with holes on the side to enable access to the external components. A 3D-printed mount is used to keep the Raspberry Pi from moving around

This is really important: if the Raspberry Pi or any of the electronics get wet or even very damp, they will fail or start to corrode

Above These glands are used to fit the cables snugly and prevent water from getting inside the box

STEP 1

KEEP IT DRY

Find two waterproof enclosures: one larger one for the Pi and the breadboard or HAT, and another smaller one for the BME280 sensor. The larger box should have a couple of holes for the RJ11 cables connecting the wind and rain sensors, and for some long wires to the BME280.

If using the recommended enclosures (see p70), you can use this 3D-printable mount (magpi.cc/aadkGv) to secure the Raspberry Pi inside the larger box, and this one (magpi.cc/rGSAQc) to hold the BME280 sensor in the smaller one.

STEP 2

SEAL IT UP

Use short self-tapping screws to secure the mounts into the holes and/or grooves at the back of the larger box.

In order to get representative readings for ambient temperature and humidity, air needs to circulate around the BME280 sensor. Remove both hole covers from one side of the smaller box. You can then pass the wires for the sensor up through one hole. Make sure you mount this box outside with the holes facing downwards so that rain cannot enter this way.

Use waterproof nylon cable glands to prevent moisture entering the enclosure through the holes used for the cables.

STEP 3

PLUG THE HOLES

The larger recommended enclosure has holes on all four sides that are sealed with rubber plugs. Use three of these holes along the bottom of the box to provide an escape route for your cables. Use an M16 cable gland in each of the two outer holes and pass the cable for the rain gauge through one and the cable for the wind sensors through the other.

Use the larger M20 gland for the centre hole and feed the power cable, DS18B20 probe, and the wires for the BME280 sensor through.

STEP 4

CRACKIN', GROMMET

The hole in the M20 is quite large, so you should pad the cables to ensure a tight fit. A 3D-printable grommet is available at magpi.cc/PrfCnV. Use two grommets rotated at 180 degrees to each other so that there is no gap all the way through.

STEP 5

MOUNTED OUTSIDE

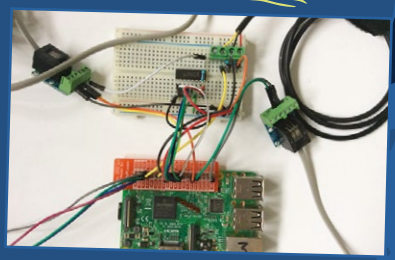
You could mount your station on a wall, rooftop, fence, or even on a plumbing pipe stuck in the ground. The larger box can be installed inside, to help keep it dry and allow easier connection to power and networking. However, the various cables for the external sensors (rain gauge, wind vane, anemometer, and BME280) will then all need to be routed inside, so this may involve a bigger hole in an external wall. Mounting everything outside means you only have to supply power to the weather station (assuming you are using wireless connectivity for data transfer).

LOCATION, LOCATION, LOCATION

It is not possible to provide specific instructions for mounting your station, as the exact method will depend on your particular location and environment. However, here are some tips for a couple of aspects of the process that should help you get started:

- **Installing your Weather Station outside: wind sensors (magpi.cc/Pxbabi).**
- **Installing your Weather Station outside: connecting to WiFi (magpi.cc/jujoYP).**

You may not be able to find an ideal location. Perhaps trees block the wind, or the rain gauge is partially sheltered by an overhang. Don't worry, just install your weather station anyway.



MAKE A WEATHER STATION HAT

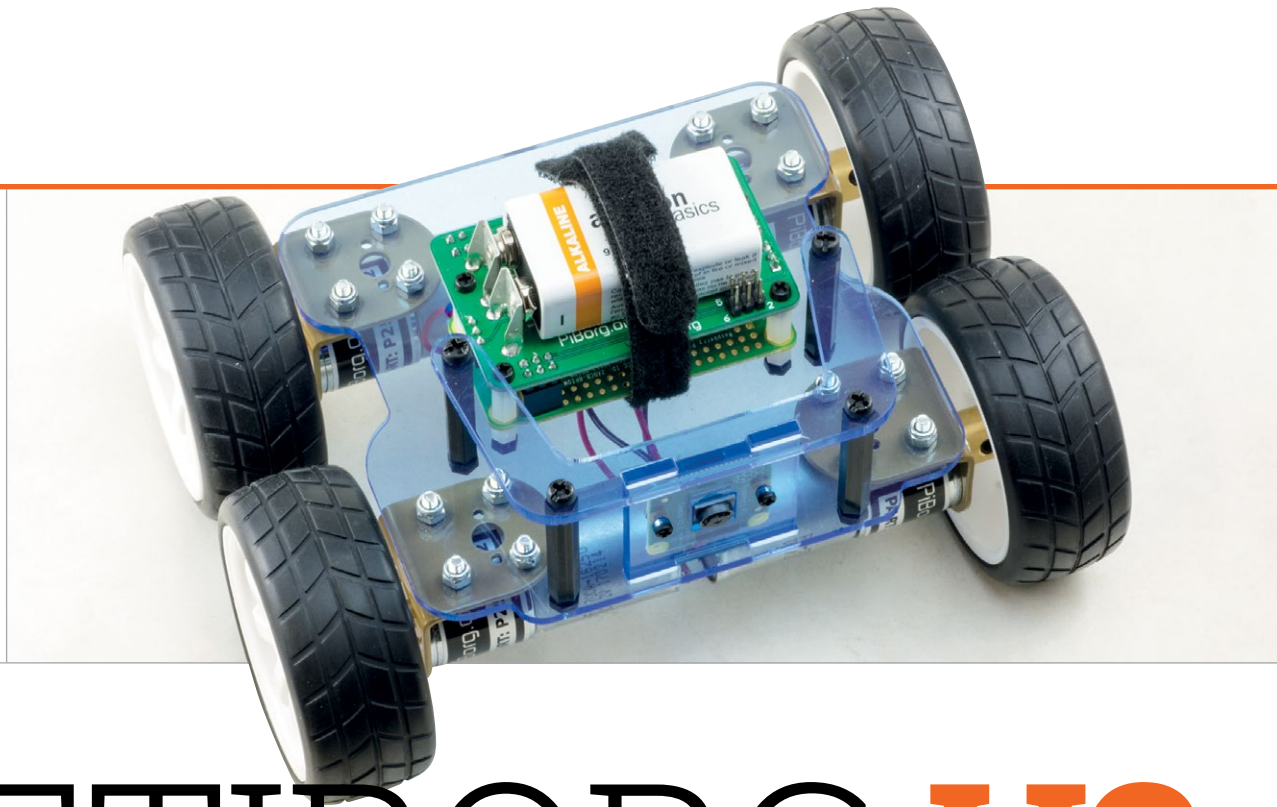
You should now have a working weather station prototype on a breadboard. For a more robust, long-term installation, or if you don't have room for a breadboard in your enclosure, you can construct a weather station HAT (hardware attached on top) for your Pi. This will involve some soldering – if you've never soldered before, Raspberry Pi has a great resource plus video tutorial to get you started: magpi.cc/VKGfig.

Note that to really be able to call it a HAT, your board should include a programmed EEPROM. The Adafruit Perma-Proto HAT kit does come as a variant with an EEPROM, so this is definitely something you could do, although we aren't covering the procedure in this guide. A good place to start is this GitHub repository (magpi.cc/ILpFCS), or this article (magpi.cc/iAEnXI).

If you want to turn the components into a HAT, then follow these instructions: magpi.cc/INdatp.

FINAL ADVICE DON'T FORGET

- **The rain gauge needs to collect rain.**
- **The anemometer and wind vane need to be in the wind.**
- **The smaller BME280 box needs to breathe – try to avoid situating it in direct sunlight.**
- **The weather station needs to be connected to power, and to a network (wirelessly or via an Ethernet cable).**



Maker Says

"YetiBorg v2 is a great kit to get started with Raspberry Pi robotics!"
PiBorg

YETIBORG V2

Rob Zwetsloot looks at the new YetiBorg robot kit. Does its diminutive size make it better for beginners?

Last month in *The MagPi* we took a look at the DiddyBorg v2 robot kit from PiBorg. This relatively large kit from the PiBorg team is a fantastic, robust robot kit that is fantastic for veteran Raspberry Pi robot builders wanting something a bit more advanced to play with. We don't think it's the best kit to choose for beginners, though, which is where the YetiBorg v2 comes in.

In comparison to the DiddyBorg, it's pretty small, although it's definitely not the smallest Pi robot kit out there. Unlike other beginner-friendly robot kits, it includes all the high-quality parts and chassis you'd expect from a PiBorg kit. This quality comes at a price, though, and at £160 it's quite a bit more than your classic robot starter kit.

Construction is pretty simplified, with a fantastic step-by-step guide that takes you through the entire build process. There's no soldering involved as it comes complete with pre-soldered motors and a Raspberry Pi Zero with a pre-soldered GPIO header. We received our YetiBorg fully constructed, in fact, but we estimate you'd be able to build it in under an hour, and the software won't take you long to sort out either.

This kit comes with a ZeroBorg, a quad motor controller designed with the Pi Zero in mind. While it may be smaller than the ThunderBorg controller used in the DiddyBorg, the ThunderBorg is only able to control two (or two sets of) motors at a time. This means the YetiBorg is truly a four-wheel-drive robot. Like the ThunderBorg, you can stack ZeroBorgs to add more motor controls if you wish, and while it is designed around the Pi Zero form factor, there's no reason you can't use it with a full-sized Raspberry Pi if you so wish.

High performance

The YetiBorg comes with example scripts to get you started, including a remote control script using a game controller, a web interface that lets you see through a mounted Pi camera (not included), automated scripts, and more. You can use these to learn how the robot works and then cobble together your own scripts so the robot will do as you wish.

While the ZeroBorg code is still quite complex as per the ThunderBorg code, it's a bit easier to understand overall. It's no GPIO Zero but it's still readable, although we suggest that you

Related

CAMJAM EDUKIT 3

A beginner's kit that's slightly more suitable for true novices. The price is very attractive as well.

£18 / $24
magpi.cc/28KDW9S

magpi.cc/TADbiy

£160 / $224

already have a basic grasp of Python before tackling it.

Still, it's quite fun to have the YetiBorg trundling around using a game controller. It's not super-quick either, so you won't have any problems with it running off your table or smashing into walls, and the extra functions with the example remote-control script – such as a button to reduce your speed – give you an idea of how you can program and control the YetiBorg.

> "With its ZeroBorg quad motor controller, the YetiBorg is truly a four-wheel-drive robot"

Friendly for beginners?

One of the selling points of the YetiBorg over the DiddyBorg is that it's for people getting into Raspberry Pi robotics; however, we're not quite convinced that it is. At least, we wouldn't say it's well suited to kids or teens looking to learn about the Pi, computing, and robotics with a Raspberry Pi robot kit. The barrier for entry with the code is just a little too high compared to other robot kits we've used in the past.

However, we do think it's quite well suited for adults confident in their computing skills who want to try their hand at robotics with a Raspberry Pi. The simple setup allows anyone to get it up and working pretty quickly, while the example scripts help you to become familiar with the ZeroBorg libraries. Whereas kids' robot kits tend to be a bit cheap and disposable, the YetiBorg works well as a basic platform to then grow from, rather than throwing it away when you get your new robot kit.

As for younger makers, while it may not be a good first kit, it could perhaps make an excellent second kit to really learn about the ways of Raspberry Pi robotics.

Last word

More of an adult beginner's kit than a young maker's kit, this robot is still a great product from the PiBorg team for those wanting something smaller than the DiddyBorg.

★★★★☆

magpi.cc/mmLdQm

£80 / $107

Maker Says

"Ads can be blocked on any device and even in apps"

Pi Supply

PI-HOLE

THE NETWORK-WIDE AD BLOCKER

Block annoying internet adverts using this pre-built Pi kit

Billed as a black hole for internet adverts, Pi-hole is free and open-source software for the Raspberry Pi. While you could create your own Pi-hole server, Pi Supply's kit comes pre-built and is almost a plug-and-play device. Inside the sleek, branded black case – also available separately for £15 – is a Pi 3 and 16GB microSD card with Pi-hole pre-installed (in Raspbian). You also get a power supply, plus HDMI and Ethernet cables. An online setup guide is provided at **pisupp.ly/pihole**.

Initial setup involves wiring the Pi-hole device to your router and connecting it to a TV. While Pi-hole is pre-installed, you need to set it up for your network. A Terminal command brings up a setup wizard, which prompts you to input your router's IP address and set a static IP for the Pi-hole. You can then run the device as a headless server, with a wired or wireless connection to the router.

Enable ad blocking

The main way to enable network-wide ad blocking is to set your router's DNS address to that of the Pi-hole. However, some routers (including ours) don't permit users to change the DNS setting. One alternative is to disable the router's DHCP server and enable one on the Pi-hole. This seemed a little daunting, so we opted to instead set the DNS address to Pi-hole's IP manually on each device – a more flexible option.

We soon encountered a problem, however: no webpages were loading! After a lot of Googling and head scratching, we eventually found a setting in the Pi-hole's web interface: changing the DNS listening behaviour to 'all interfaces' instead of 'etho' did the trick (since our Pi-hole wasn't wired to the router). The web interface is also very handy for seeing how many web queries have been blocked, viewing a query log, and black/whitelisting any that should be blocked/unblocked.

Related

PI-HOLE FOSS

Alternatively, to build your own Pi-hole server with an existing Raspberry Pi, you can install the free and open-source Pi-hole software.

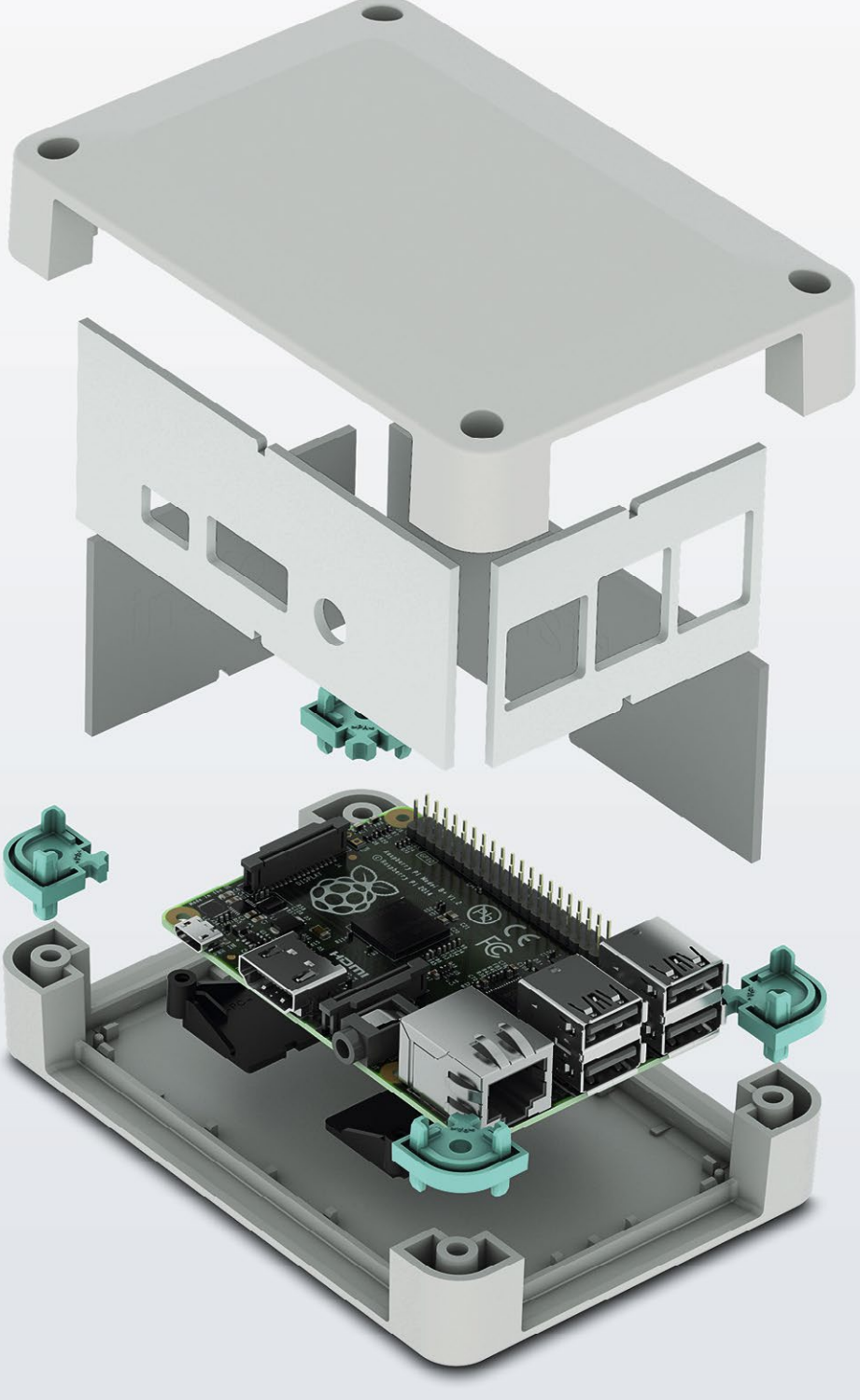
Free

pi-hole.net

Last word

While setup wasn't quite as easy as we'd hoped – including having to set a new password to log in to the web interface – the Pi-hole device worked really well on the normally ad-heavy sites we tried. Yay, no more autoplay video ads!

★★★★☆



Raspberry Pi 3 B+ finds a new home

Versatile enclosure for Raspberry Pi 3 B+

The new UCS-RPI Universal Case System is compatible with the recently launched Raspberry Pi model 3 B+. It has pre-milled side walls for easy access to the I/O and power inputs and is available in black or grey and two sizes. Complete with glue dot location posts to secure the single board computer to the case.

For additional information call 0845 881 2222 or visit

phoenixcontact.co.uk/UCS-RPI

PHOENIX CONTACT
INSPIRING INNOVATIONS

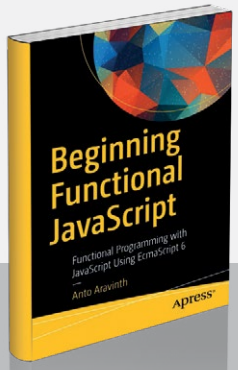
RASPBERRY PI BESTSELLERS

FP TO WEB

ClojureScript, along with JavaScript's functional features, enables more robust and responsive web apps in a 'serverless' world.

BEGINNING FUNCTIONAL JAVASCRIPT

Author: Anto Aravinth
Publisher: Apress
Price: £23.99
ISBN: 978-1484226551
magpi.cc/RxKNXL



If you're an OO programmer struggling with the concepts of functional programming, Aravinth's explanations should help make things clear. A second edition with better editing would be welcomed, but the author's knowledge and enthusiasm will carry you past grammatical errors.

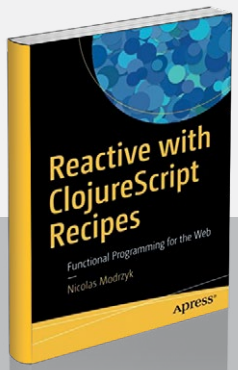
QUICK CLOJURE

Authors: Quick Clojure
Publisher: Mark McDonnell
Price: £22.99
ISBN: 978-1484229514
magpi.cc/fEJhMU

If you'd rather get the basics of Clojure under your belt before tackling ClojureScript (or a Clojure web server-based project), this concise guide is a truly enjoyable introduction to all the bits of the language that you'll really need.

REACTIVE WITH CLOJURESCRIPT RECIPES

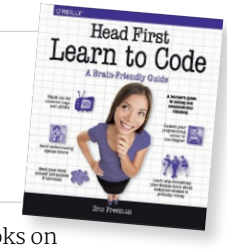
Author: Nicolas Modrzyk
Publisher: Apress
Price: £31.99
ISBN: 978-1484230084
magpi.cc/xXjmjs



ClojureScript is a Clojure that compiles to JavaScript and, as Modrzyk shows here, works well for functional reactive programming (FRP). The recipe book approach works well – allowing for beginner and relatively advanced topics to rub along well together.

HEAD FIRST LEARN TO CODE

Author: Eric Freeman
Publisher: O'Reilly
Price: £39.99
ISBN: 978-1491958865
magpi.cc/lTnVbT



There are many books on learning to code with Python, and several of them are actually fun – and we've really enjoyed reviewing them – but the latest book in O'Reilly's excellent Head First series stands out. The first chapter is about 'Thinking Computationally', and combines an introduction to what programming languages are and how they work, with an easy and practical introduction to some Python code. Nicely done, and very beginner-friendly.

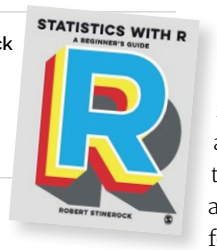
Variables are introduced with a dog age calculator program, then Booleans and control structures are explained with Rock, Paper, Scissors. After lists, functions, and a look at the bubble sort algorithm, there's a substantial project which puts everything together, and involves slicing strings and lists to measure the readability of texts – playing to Python's strengths.

Six more chapters build on readers' achievements and give the skills and techniques needed for programming, from making reusable code and working with the file system, via recursion, to interacting with web APIs and building a graphical user interface. From using Turtle graphics to represent location data from the International Space Station, to canine humour in the OOP chapter, Freeman keeps this fun and engaging. Great for adults who want to learn to code, but also recommended for teen learners.

Score ★★★★★

STATISTICS WITH R: A BEGINNER'S GUIDE

Author: Robert Stinerock
Publisher: Sage
Price: £100
ISBN: 978-1473924895
magpi.cc/jKQuZz



A student textbook which introduces statistics and analytical skills through the R language, reflecting the change in teaching even dry and conceptual topics like statistics. Indeed, knowledge of R is close to essential for those looking to learn statistical methods and start a career in finance, pharmaceuticals, or even the civil service. This book is differentiated from many R introductions in being solidly grounded in teaching everything a first-year student of statistics needs to know.

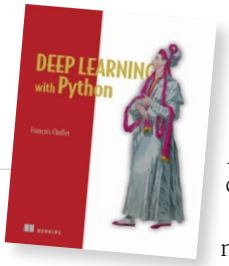
Tabular, graphical, and numerical methods in statistics lead on to problems in probability that depend upon them, and a look at everything from sample space to Bayes' theorem. Discrete and continuous probability functions lead on to binomial probability distribution – concepts again reinforced with the exercises. More advanced topics, such as hypothesis testing and multiple regression, are introduced after the appropriate foundation knowledge. A very comprehensive introduction.

The formal textbook approach carries the penalty not just of a certain dryness that might make difficult reading for those who've become used to programming books leavened with humour, but also an eye-watering textbook price, although it's also available as a paperback for £34.99. For those not constrained by budget, this is a good guide indeed – clear, well structured, and featuring well chosen examples and exercises.

Score ★★★★½

DEEP LEARNING WITH PYTHON

Author: François Chollet
Publisher: Manning
Price: £39.99
ISBN: 978-1617294433
magpi.cc/cvqSKy



François Chollet is the originator of the Keras deep learning library, and wrote this book to address the need for a course to "simultaneously cover fundamentals of deep learning, Keras usage patterns, and deep learning best practices." This user-friendly, open-source, high-level neural networks API was developed "with a focus on enabling fast experimentation," enabling quick learning and testing of ideas.

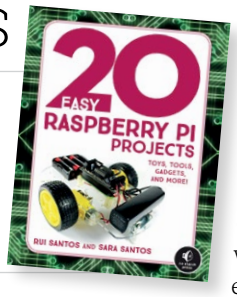
The Keras project has excellent documentation, but Chollet's book provides a great introduction to machine learning, neural networks, and best practices for deep learning. Example-led explanations are clear, and the total beginner to the field is quickly learning quite advanced topics without fear – with examples to take to real-world problems. Appendices cover not just installation, but running examples in Jupyter notebooks with cloud-backed GPUs.

It's less maths-heavy than many deep learning books, but for most that will be a selling point. Certainly, Chollet's explanations and examples will give you most of the knowledge you need, with a lot less mental strain, and readers prepared to tackle some of the more mathematical books may benefit first from some practical experience found here. Broadly recommended for anyone looking to dive into the exciting field of machine learning.

Score ★★★★★

20 EASY RASPBERRY PI PROJECTS

Authors: Rui Santos & Sara Santos
Publisher: No Starch
Price: £21.99
ISBN: 978-1593278434
magpi.cc/MEAAvk



For beginners fascinated by the possibilities of fun, simple projects on the Raspberry Pi – or educators without an electronics background, looking to learn and get teaching examples – this colourful and well-laid-out introduction could be ideal. The 20 practical project tutorials include details of cost, time needed, parts list, basics electronic knowledge (practical, not theoretical), and what you can do to take it further.

Projects start with LED basics, then add mini screens (Pong, weather forecaster), sensors, camera (burglar detector with photo capture), web apps, and toys and games. One of the highlights in the latter section is the Digital Drum Set. The photography is extremely clear, helping beginners to follow the circuits easily.

Along the way, Python code is given, and explained line-by-line where appropriate. There's enough Python intro text to explain what you're doing, but it's not a full coding intro – just the right amount of information to aid those who want to get on with making things, and learn as they go. There's also a Scratch game, controlled by two push-buttons. It's interesting to see Scratch introduced towards the end of a book, and liberating to not be bound by the idea that graphical comes before text-based.

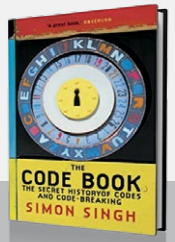
Score ★★★★☆

ESSENTIAL READING: SUMMER READING ESSENTIALS

Summer means downtime – perhaps a beach holiday – and time to read and think.

The Code Book

Author: Simon Singh
Publisher: Fourth Estate
Price: £10.99
ISBN: 978-1857028898
magpi.cc/jtCAbf



An eminently readable and understandable guide to codes and code-breaking, from ancient ciphers to PGP and quantum cryptography.

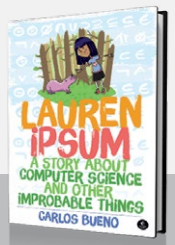
The Hardware Startup

Authors: Brady Forrest, Renee DiResta & Ryan Vinyard
Publisher: O'Reilly
Price: £27.99
ISBN: 978-1449371036
magpi.cc/IZzQNQ

Practical guide to knowing your market, funding, developing your product, and manufacturing: is your project a viable business?

Lauren Ipsum

Author: Carlos Bueno
Publisher: No Starch
Price: £13.50
ISBN: 978-1593275747
magpi.cc/Eowlip



Great to see the kids reading anything, but this (and *The CS Detective*, also from No Starch) will entertain and educate simultaneously.

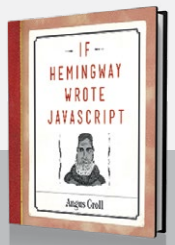
Clean Architecture

Author: Robert C. Martin
Publisher: Prentice Hall
Price: £27.99
ISBN: 978-0134494166
magpi.cc/ShtTnu

Insightful collection of software architecture advice from Uncle Bob – principles and techniques which will help any project.

If Hemingway Wrote JavaScript

Author: Angus Croll
Publisher: No Starch
Price: £15.99
ISBN: 978-1593275853
magpi.cc/xNKMLo



A coding book you could read on the beach – humour and style insights with code by everyone from Calvino to Chaucer [allegedly!].

THE MONTH IN RASPBERRY PI

Everything else that happened this month in the world of Raspberry Pi

THE CHALLENGE OF TRANSLATION

KAT LEADBETTER TELLS US HOW YOU CAN HELP **CODE CLUB** AND **CODERDOJO** TRANSLATE THEIR FREE RESOURCES

At Code Club and CoderDojo, we're always working towards our long-term mission: to start Code Clubs and Dojos in every community on Earth. As part of that, we're committed to helping as many children as possible access our resources in their native languages. The challenge? There are more than 6000 languages spoken around the world today...

When Code Club and CoderDojo began to expand outside the UK and Ireland, amazing members from across the community realised the need for translated materials and stepped forward to offer their help. Thanks to these volunteers, we have since translated projects into more than 20 languages, from Croatian to Chinese and Portuguese to Polish.

"Last year, I translated Code Club projects into Chinese," says volunteer Jarod Yv. "Now kids can read the project articles by themselves and follow the tasks to create their own programs step-by-step. They get more coding practice and improve their programming skills rapidly."

With our increasing number of translations, volunteers are also motivated to make the most of the projects they have worked on. Code Club volunteer Marcus is planning to start using Welsh translations in his club, Layla will use Farsi translated projects in Iran, while Venancio is keen on making Raspberry Pi better known in Colombia.

Our volunteer translators come from all backgrounds, professions, and walks of life. No technical or coding knowledge is needed! Translating is also a great way to get involved if you have language skills but don't have the time to volunteer at a regular Code Club or Dojo. Translators give a couple of hours of their time each week, but it's up to you when you decide to work on translations. Depending on your English skills, we can also match you up with tasks that best suit your ability and confidence level.

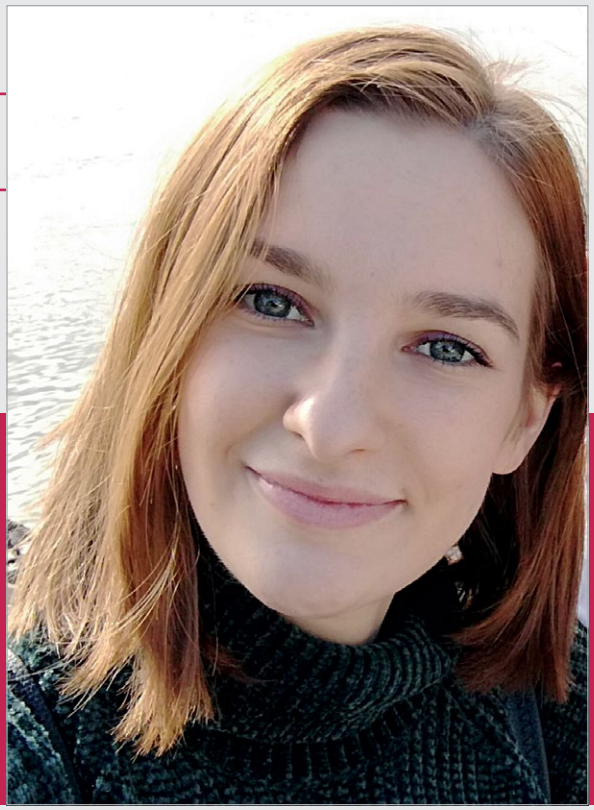
How does translating work?

To make our translation process as easy and efficient as possible, we use the latest technology. Machine translation uses artificial intelligence to translate phrases automatically, without the help of humans, and translation memory saves all previous translations and suggests them to translators who are working on something similar.

Translators use an online platform called Crowdin, which combines these technologies in one place and provides an easy and intuitive editor for translators to work in. Projects are first translated with machine translation, after which a volunteer checks translation and makes any necessary changes. The project then goes through two other steps (review and test) to help us make sure that the quality of the final translation is excellent. At each step of the process volunteers are supported by our amazing Translations Manager, Nina, and there is also a community of other translators ready to give you a hand.

MAJA MANOJLOVIC

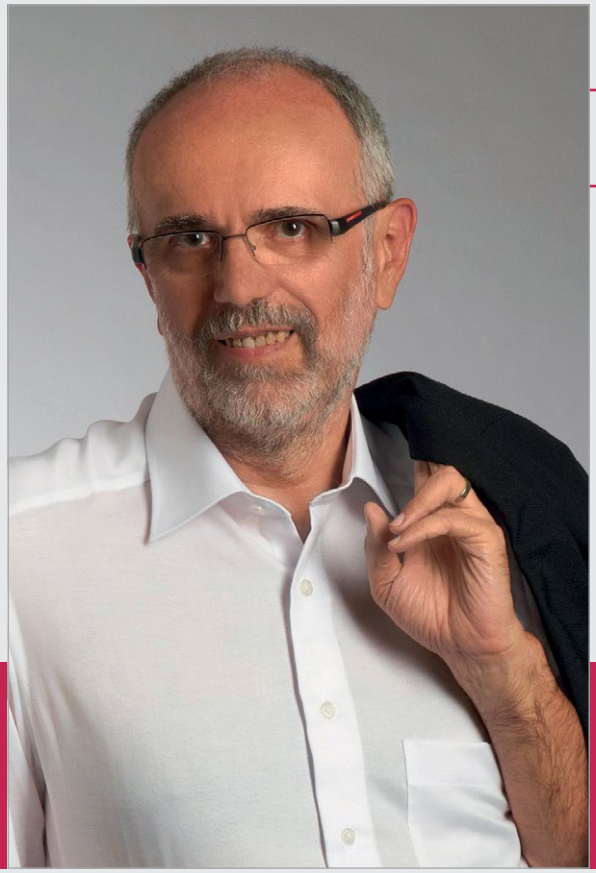
Language: **Croatian**
Number of projects translated or reviewed: **23**



I'm a student at the University of Rijeka in Croatia. My college professor suggested I help translate Code Club projects as a part of my M.A. thesis – since I am majoring in English and Informatics, it was a great opportunity to connect both my majors and gain more experience in translating!

What advice would you give to others thinking about translating?
I would encourage anyone who has any interest in translating or coding to get involved, because it is a great way to engage in something fun and new. You will learn so much, explore new things, engage in interactions with other volunteers – the list is endless! The fact that you are helping children all over the world to learn coding is also a reward in itself!

Favourite project:
Rock, Paper, Scissors, where you learn to code this well-known game. I've learned the basics of Python whilst translating and discovered a lot of fun and interesting things you can do with code. I even printed out the materials to go through them once again, this time as a student learning to code in Python!



KARL SCHUH

Language: **German**
Number of projects translated or reviewed: **40**

How did you start translating?
I think through *The MagPi*. Late in 2016 I had a go at translating the Code Club project 'Happy Birthday' and last year I got an email from Code Club and decided to get involved again. My main motivation was to go over the basics of using a Raspberry Pi and learn about some simple exercises which could be used for teaching adults, if the need for it came up at my job.

Favourite project:
I like Raspberry Pi's various projects for lighting a traffic light, since it can be expanded to more and more functions.

What advice would you give to others thinking about translating?
I really like to translate. It is quite easy to do if you are a bit of a techie and it gives you a good feeling to bring children an interesting and creative way to spend their time. My advice: just start doing it! It is a nice community and all of us are willing to help newbies.

What have you found most rewarding about translating?
Translating pushes me to polish up my knowledge of the English language. I understand uncommon phrases and words I didn't know before.

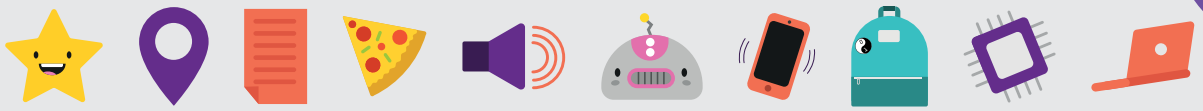
COR GROOT

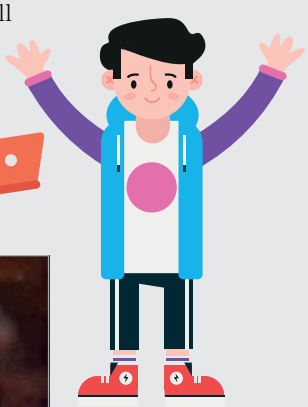
Language: **Dutch**
Number of projects translated or reviewed: **52**

How did you start translating?
Since my retirement I have volunteered at a hackerspace where we introduce students from primary schools to everything from soldering, to coding with Scratch and robotics. For the computer-related workplaces, I developed all the lesson packages and found how difficult coding in English can be for Dutch youth. I started looking at how I could contribute to changing that.

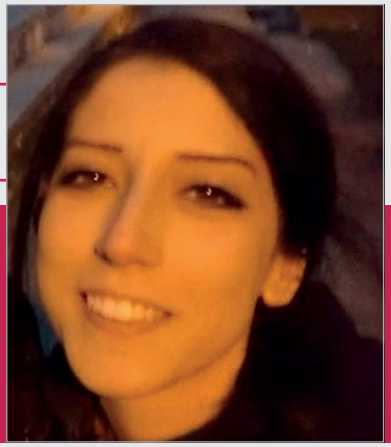
Because I play a lot with Raspberry Pi and Arduino, I soon came into contact with the Raspberry Pi Foundation, as well as Code Club and CoderDojo. That led me to join the translation group. At the moment I have translated and reviewed various projects for the Raspberry Pi, Scratch, HTML & CSS, and Python.

What advice would you give to others thinking about translating?
Translating is easy with the help of various translation sites on the internet, and we regularly chat within our translation group about the best translation for a sentence or subject. If you want to help us to introduce young people to coding, then join us, choose a project that suits you, and start translating! You're helping children, you will learn a lot yourself, and most importantly: it's fun!





SILVIA CAPONIO



Language: **Italian**
Number of projects translated or reviewed: **45**

I'm a 23-year-old language enthusiast from Bari, southern Italy. At the moment I'm studying for my Master's at the University of Sheffield, specialising in – you guessed it – Translation Studies.

Favourite project:
Boat Race has a special place in my heart, because it was the first one I translated, and the one I spent the most time on. It was a whole new world for me!

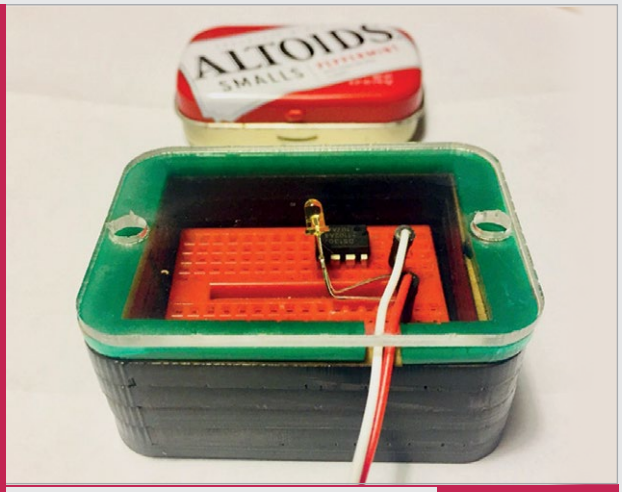
How did you start translating?
When my professor said Raspberry Pi was looking for volunteer translators, I immediately signed up. Their projects introduce young people to computing and digital making, and develop their problem-solving skills in a fun way – something I strongly support. Through the projects I found out about Code Club and CoderDojo, and I found myself immersed in an amazing, positive community of people who work hard and collaborate and communicate with each other every day.

What advice would you give to others thinking about translating?
Translating and localising technical texts for a younger audience is an incredibly fun, challenging, compelling task. The translation process seems complicated at first, but the guidelines are straightforward and you can always ask for help! My advice for any prospective volunteer who wants to join this brilliant community is – have fun! Don't be afraid to ask for advice: everyone is there to help you!

To become a volunteer translator, head to **rpf.io/translators** and fill in your details. With your help, many more children around the world could discover a world of coding and digital making!

CROWDFUND THIS!

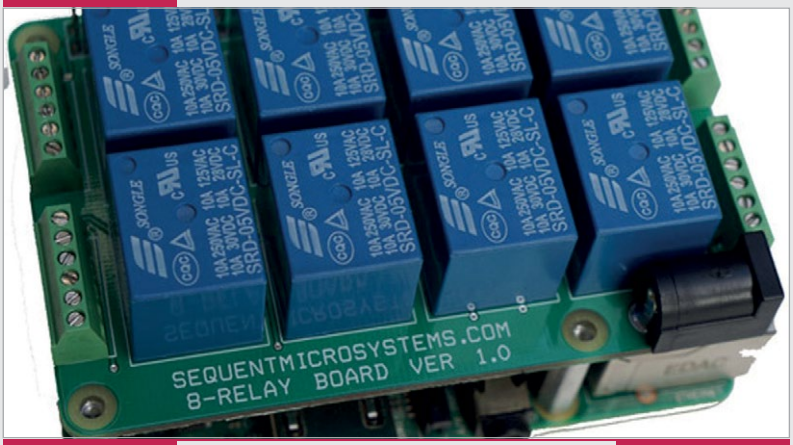
The best crowdfunding hits this month for you to check out...



kck.st/2LIOPsA

NIBBLER CASE 170

This one is not specifically for a Raspberry Pi, but is a great way to organise your breadboard circuits for Pi projects. The case is made of laser-cut acrylic or wood depending on your choice, has magnetic catches, and comes with a little breadboard that fits perfectly inside.



kck.st/2LSkZlx

STACKABLE 8-RELAY CARD

A spin-off of the Mega-IO expansion card (**kck.st/2eLc4no**) that removes a lot of the functionality in favour of just including the relays. They can be stacked easily, which allows for greater control of many more devices than what you can do with just the GPIO pins on their own.

BEST OF THE REST

Here are some other great things we saw this month

PI-POWERED GRADUATION CAP

Apparently this is Reddit user CHRISFRED's first Pi project – they've installed LEDs into the top of a cap to show during graduation. It cycles between different messages and images – hopefully it's controllable so that when it's thrown into the air, the perfect picture can be taken. Especially as it will probably break as it crashes to the floor.

magpi.cc/PZQLaP

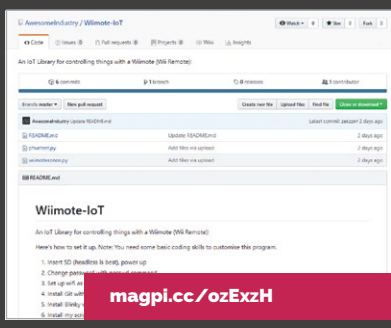
magpi.cc/PDBBzr

LAWNMOWER ROBOT

If you've not heard of them before, the Let's Robot team put up video streams with their robots which the viewers can control. Now you can steer a Pi-powered robot that cuts the lawn. This is frankly amazing – the grass will never have been so thoroughly cut!

WIIMOTE-IOT

We got an email from Isaac, 14, of Awesome Industry to tell us about his great new Pi Wiimote-IoT project on GitHub. You can use it to control Phillips Hue lights and Sonos systems with a Wii Remote, and we think that's pretty cool.



magpi.cc/ozExzH

COMMUNITY PROFILE

BRIAN CORTEIL

When it comes to robots, **Brian Corteil** is a master maker

Brian

Category: Robot Maker Extraordinaire

Day job: Security Engineer

Website: corteil.co.uk/Blog

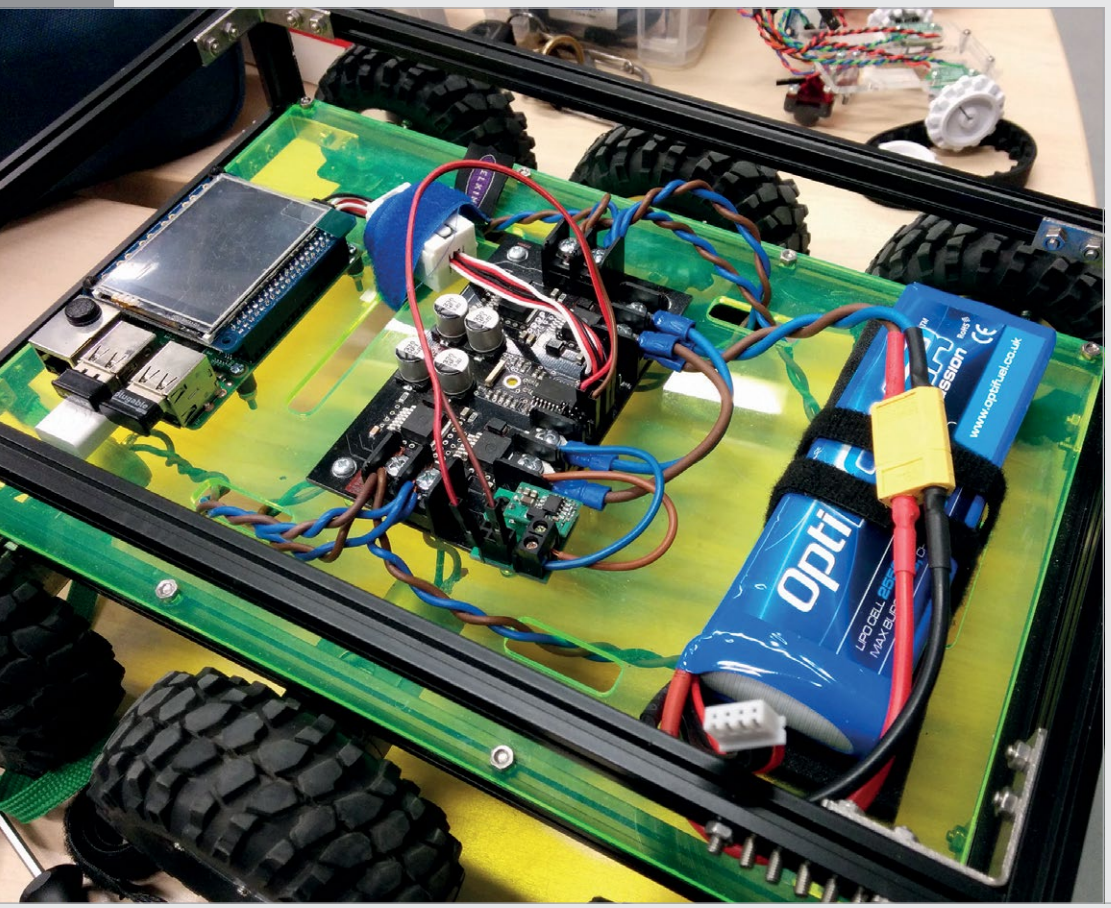
twitter.com/CannonFodder

core-tec.co.uk

If you've read *The MagPi* before, you'll recognise Brian Corteil's name. From tutorials and events, to online discussion and Pi parties, it's likely you'll bump into him with a robot in one hand and some other mad creation in the other.

Trained as an electrician, Brian's interest in electronics came into play with the release of the Arduino and, later, the Raspberry Pi, after he was able to obtain one of the first 10 000 Model Bs. By day, he's an integrated security systems engineer, installing service access control, CCTV, and intruder alarms. And by night, Brian is a keen and successful robot builder.

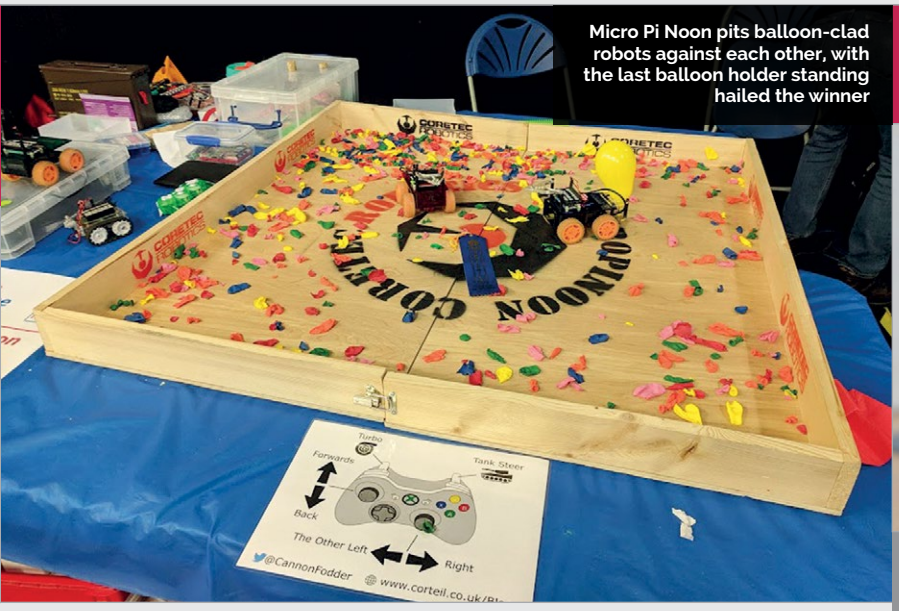
The annual Pi Wars challenge is where Brian's robot-making skills truly shine. In fact, if you were to ever bump into Brian at his local Cambridge Makespace, he'd likely be tweaking his latest robot buggy or building new track for the main event. And the time and effort put in definitely has its rewards, with Brian taking first place in several of the pro courses of this year's Pi Wars, missing out on overall first place by a mere one point.



Below Controlled by a PlayStation 3 gamepad, Revenge is a six-wheeled A3 class robot that can cope with a host of different surfaces

Robotics made easy

Outside of Pi Wars, Brian is also the head of Coretec Robotics, responsible for the Tiny 4WD, a mini robot originally designed for *The MagPi*'s 'Build your own Pi Wars robot' feature in issues 51 and 52. The success of the robot turned 4WD into a consumer project kit that avid makers can buy and build themselves using laser-cut parts, a Raspberry Pi Zero W, a handful of electronics, and the Pimoroni Explorer pHAT. "I have to thank the salty sea dogs of Pimoroni for kicking me up the bum to get Tiny out in the wild," he jokes



Micro Pi Noon pits balloon-clad robots against each other, with the last balloon holder standing hailed the winner

when talking to us about Coretec. "Over the next year or so, I will be releasing helpful breakout boards for making robotics and making a little easier."

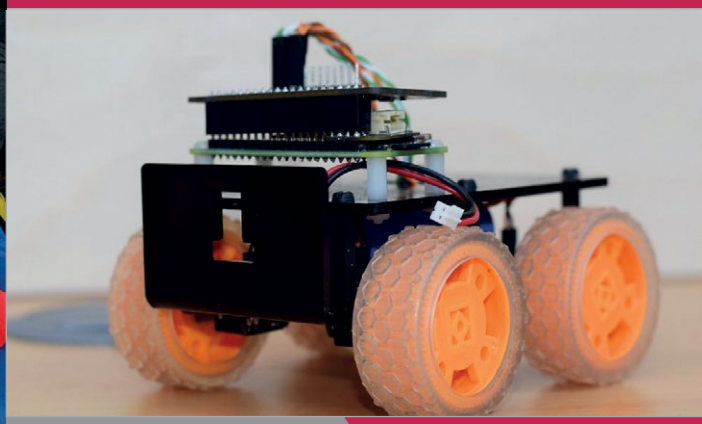
Brian also tries to attend as many events as possible, from Maker Faires to Raspberry Jams, and can often be found demonstrating Micro Pi Noon, based on Michael Horne and Tim Richardson's Pi Wars Pi Noon challenge. Two competitors battle to pop the balloons attached to each other's robots; the last to retain an inflated balloon is the winner. "Thousands of balloons have been sacrificed to the maker gods," he admits.

When it comes to discussing his project builds and robots, Brian is first to list the people to whom he's beholden. "I like to thank the people who have supported me in my adventures in making, first of all the members of Makespace for helping me turn my crazy ideas into working things," he begins, before mentioning organisations such as Pimoroni, Tim and Mike (the minds behind Pi Wars), and makers Robert Karpinski and Mark Mellows "for checking my electronic designs and answering my silly questions and their help in being awarded a blue ribbon at this year's Maker Faire UK in Newcastle."

> "Thousands of balloons have been sacrificed to the maker gods"

Brian won a blue ribbon at this year's Maker Faire Newcastle for his robot builds, including the six-wheel Revenge and A6-sized Tiny

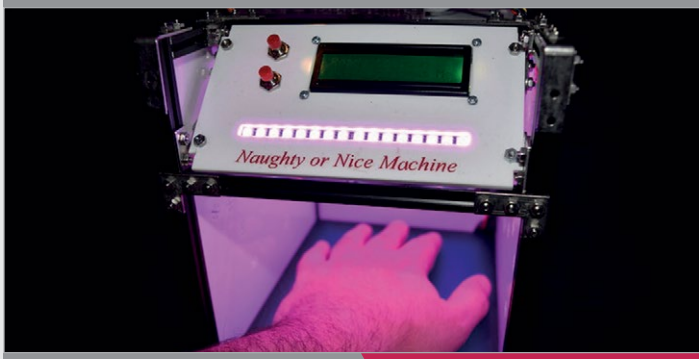
HIGHLIGHTS



magpi.cc/muyKWe

ROBOTS

When it comes to robots, Brian is the Cambridge maker extraordinaire. From the A6-sized Tiny to the whopping Revenge, his robots dominate competitions and Raspberry Jams alike.



magpi.cc/xpLluo

NAUGHTY OR NICE

The Naughty or Nice machine determines your level of naughtiness in time for the arrival of Santa. And for some 'unknown reason', smaller hands tend to boost the level of niceness.

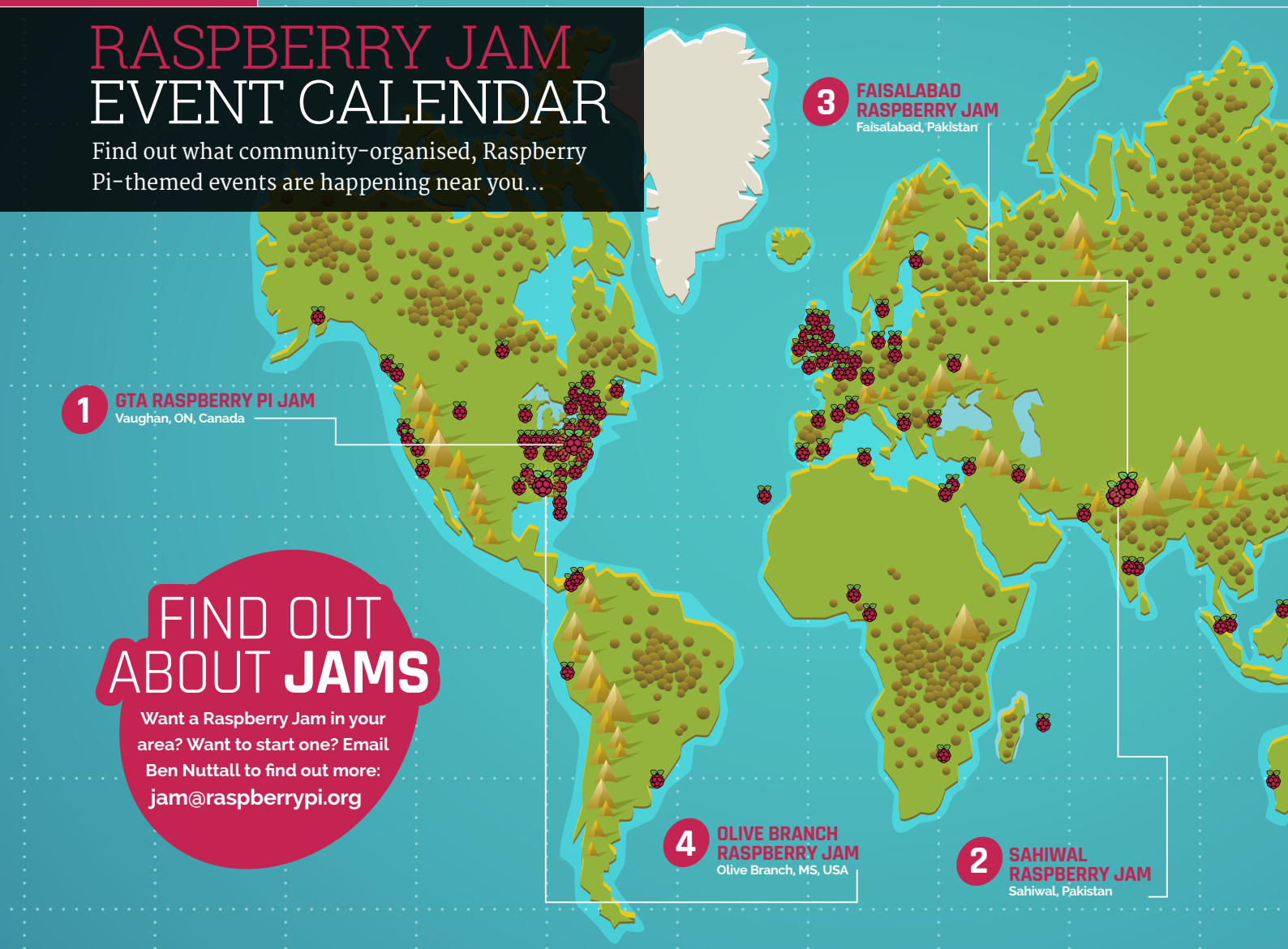
magpi.cc/2cotnva

DIGITAL ZOETROPE

Brian's digital zoetrope uses 12 Adafruit OLED screens connected to a Raspberry Pi. Each screen displays a single frame of a video; when the device is spun, the video can be viewed through gaps in the outer wall. A modern take on a mid-19th century classic.

RASPBERRY JAM EVENT CALENDAR

Find out what community-organised, Raspberry Pi-themed events are happening near you...



FIND OUT ABOUT **JAMS**

Want a Raspberry Jam in your area? Want to start one? Email Ben Nuttall to find out more: jam@raspberrypi.org

1-5 HIGHLIGHTED EVENTS

1 GTA RASPBERRY PI JAM

When: Thursday 5 July
Where: Civic Centre Resource Library, Vaughan, ON, Canada
magpi.cc/CMhJUg
Makers, tinkerers, coders, and lovers of playing with technology unite for this free Jam!

2 SAHIWAL RASPBERRY JAM

When: Monday 9 July
Where: E-Rozgaar Center, Sahiwal, Pakistan
magpi.cc/yHzkXy
A Maker.pk-organised Raspberry Jam in Pakistan, one of two this month, and free for all to attend!

3 FAISALABAD RASPBERRY JAM

When: Thursday 19 July
Where: University of Agriculture, Faisalabad, Pakistan
magpi.cc/XcpdWu
The second major Pakistani Raspberry Jam this month, which is also free for everyone.

4 OLIVE BRANCH RASPBERRY JAM

When: Saturday 28 July
Where: B.J. Chain Public Library, Olive Branch, MS, USA
magpi.cc/WxeYYR
A free family event to learn about coding and play with the Raspberry Pi.

6-8 REGULAR EVENTS

5 PRESTON RASPBERRY JAM

When: Monday 2 July
Where: Media Factory Building, Preston, UK
magpi.cc/BvJeFL
A community of people who meet to learn, create, and share the potential of the Raspberry Pi.

6 OXFORD RASPBERRY JAM: EVENING FLAVOUR

When: Tuesday 3 July
Where: Oxford Centre For Innovation, Oxford, UK
magpi.cc/XixtqF
An evening meet-up for Raspberry Pi enthusiasts! Suitable for adults and accompanied young people.

WE'VE HIGHLIGHTED SOME OF THE AREAS IN NEED OF A JAM!
CAN YOU HELP OUT?
8 YORK RASPBERRY JAM
York, UK
7 LEEDS RASPBERRY JAM
Leeds, UK
5 PRESTON RASPBERRY JAM
Preston, UK
6 OXFORD RASPBERRY JAM: EVENING FLAVOUR
Oxford, UK

7 LEEDS RASPBERRY JAM

When: Wednesday 4 July
Where: Swallow Hill Community, Leeds, UK
magpi.cc/zNNDeX
Get hands-on with digital making activities through the workshop. Plus a hackspace area to share projects.

8 YORK RASPBERRY JAM

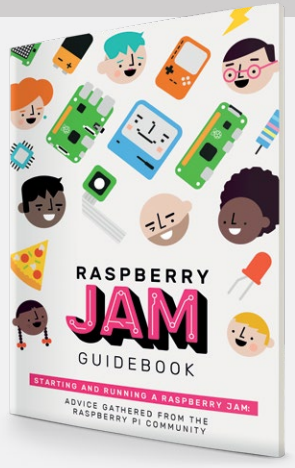
When: Monday 16 July
Where: Acomb Explore Library, York, UK
magpi.cc/hbfvvK
York Pi Jam is an event for people of all skill levels who want to learn more about the Pi.

RASPBERRY JAM ADVICE

PROMOTING YOUR EVENT

> "I use Twitter and Facebook. I've found it useful to get the message directly to schools who use Twitter, so the IT specialist can pass the message on to parents and children. I think old-fashioned posters put up around the area work well too."
>
> **Anne Carlill**
> **York Raspberry Jam**

Every Raspberry Jam is entitled to apply for a Jam starter kit, which includes magazine issues, printed worksheets, stickers, flyers, and more. Get the guidebook here: **magpi.cc/2q9DHfQ**



YOUR LETTERS

MUSICALLY INCLINED

I've used a Raspberry Pi for a few years in different electronics projects – some robots, the odd automatic light, that kind of thing – but I've become interested recently in having my music controlled and played from my Raspberry Pi.

I know there's the headphone jack and you can do digital audio via the HDMI port, but is there a way to get better audio out of the Raspberry Pi?

With that in mind, would you say the Raspberry Pi is generally good for a jukebox or party music application? Whether you're DJing or just letting it play from a playlist?
Martin

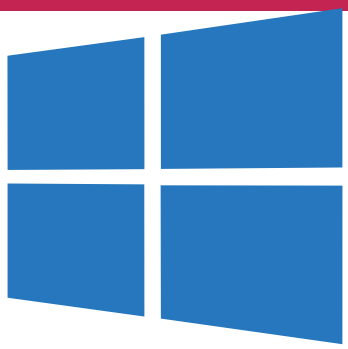
Among the most common add-ons we see for the Raspberry Pi are DACs that offer high-quality audio output designed for sound systems. A lot of them come as HATs which slot on top of the Raspberry Pi, making it all very compact, and several include RCA jacks or optical out.

There are plenty to choose from, but we quite like the Nanomesher series of DACs as they include a screen and you can get an optional 3D-printed case to fit them.

As for DJing with one – the Raspberry Pi is just a computer, so as long as you can find software you like to use for it, you can play music as you wish. We've also seen Sam Aaron, creator of Sonic Pi, do live DJing with the music creation software. There's plenty you can do with it!

The NanoSound DAC Pro is one of our favourite recent DACs for better audio

MORE THAN LINUX



While full Windows is not readily available for the Raspberry Pi, the IoT Core version is

I like using Raspbian on the Raspberry Pi, but I'm not very used to Linux and those types of operating systems. I've also seen that a lot of other operating systems for the Pi are based on Linux. Is there any way I can use Windows on my Raspberry Pi?
Emilio

As the Raspberry Pi is mainly designed for education, using something like Raspbian is a great way for people to learn about computing and coding. It's also slightly easier to get a Linux distribution to work on the Raspberry Pi due to its relatively low power and use of an ARM processor – Windows runs on x86 computers with much more powerful processors and such.

However, while you won't be getting a Windows 10 SD card like you would an installation CD for your computer, people have been figuring out how to hack Windows 10 to run on Raspberry Pi. It doesn't run particularly well but if it's good enough for you to use, we suggest taking a look into it.

VIDEO PLAYBACK

How does the playback of media on Raspberry Pi work? I have a Pi Zero that plays 1080p video just fine, but even my Pi 3 has problems with playing 4K video at 1080p resolution.

I also have issues with 10-bit video and HEVC. Why is this? Surely the more powerful Pi will play higher bitrate media like this?
Tom D

The Raspberry Pi has hardware decoding for a lot of 1080p video, which is why something like a Raspberry Pi 2 or newer model can comfortably run Kodi and most video you can throw at it. You might wonder if it's a problem with Kodi's codecs, but that's not the case either: basically the Raspberry Pi (at least the Zero), on its own without the hardware decoding, would struggle with 1080p video.

As the hardware doesn't technically support these types of video, it has trouble decoding video that requires a bit more oomph to play. It also does depend on your encoding – we've seen 10-bit encoded video looking washed-out and almost unwatchable, or just have a bit of tearing in a scene.

For now, it's better to encode video with H.264 for Raspberry Pi playback.

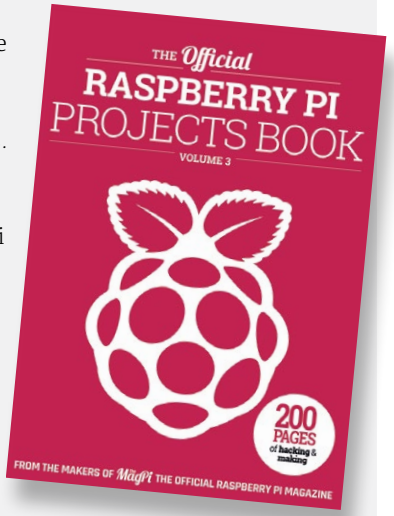
FROM THE FORUM:

PDFS FOR ALL

Here in Africa, PDFs are a lot more practical than printed magazines. I have PDFs of *The Official Raspberry Pi Projects Book* – Volume 1 and 2, but can't find Volume 3. Is this available to download? Where can I find it?
DaveSemm

Anyone having trouble finding the PDF for *The Official Raspberry Pi Projects Book* Volume 3 should head here: **magpi.cc/projects3** – you'll see a button that says 'Download Free' which lets you get the PDF for the book.

This button can be found on every issue and book's page on our website to get the PDF for free, to make it easier for people like you who cannot get the printed editions. You can even read them on the Raspberry Pi as well!


THE Official
RASPBERRY PI
PROJECTS BOOK
VOLUME 3
200 PAGES of hacking & making
FROM THE MAKERS OF MagPi THE OFFICIAL RASPBERRY PI MAGAZINE

As well as every issue of the magazine, the Projects Books are available as a free PDF!

WRITE TO US

Have you got something you'd like to say?
Get in touch via **magpi@raspberrypi.org** or on The MagPi section of the forum at: **raspberrypi.org/forums**

The Raspberry Pi Forum is a hotbed of conversations and problem-solving for the community – join in via **raspberrypi.org/forums**

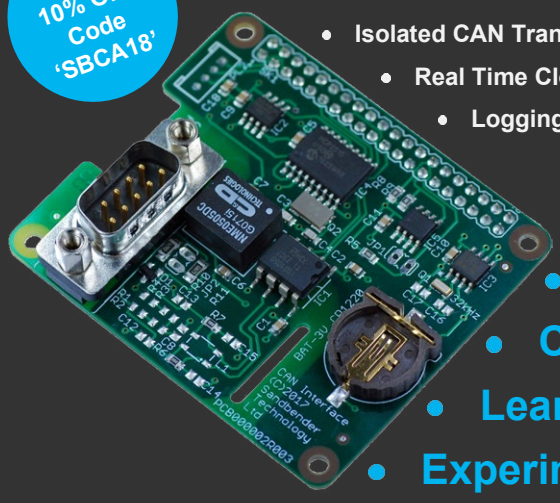
Sandbender Technology

CAN Interface pHAT

Easily turns a Raspberry Pi into a low cost CAN bus equipped controller.

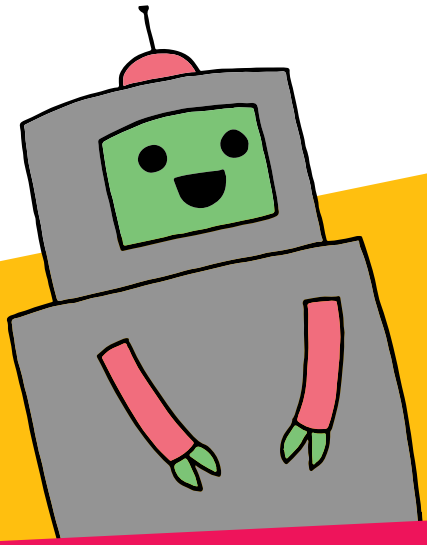
FIRST 10 10% OFF Code 'SBCA18'

- SPI MCP2515 CAN Controller
- Isolated CAN Transceiver
- Real Time Clock
- Logging Memory

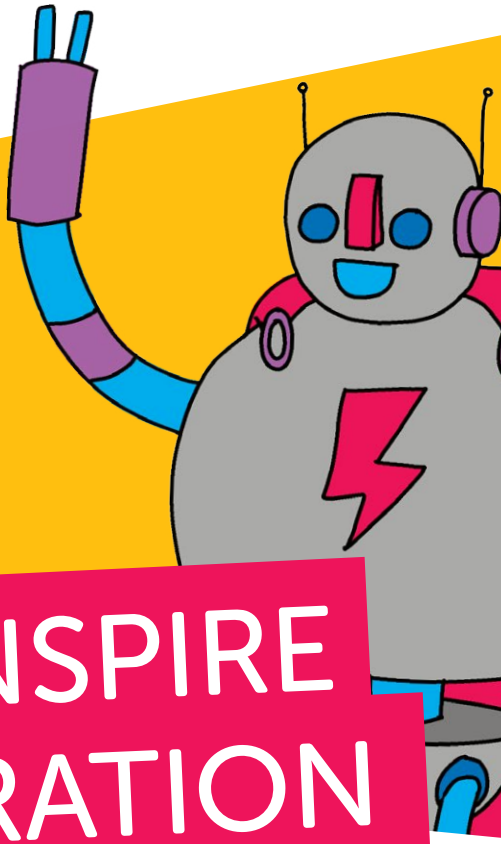


- Log
- Control
- Learn
- Experiment

www.sandbendertechnology.com



{code club}

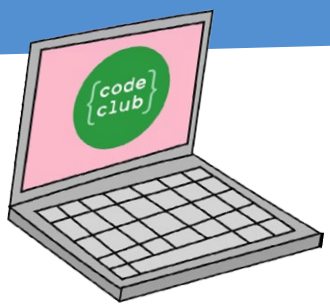


CAN YOU HELP INSPIRE THE NEXT GENERATION OF CODERS?

Code Club is a network of volunteers and educators who run free coding clubs for young people aged 9-13.

We're always looking for people with coding skills to volunteer to run a club at their local school, library, or community centre.

You can team up with friends or colleagues, you will be supported by someone from the venue, and we provide all the materials you'll need to help children get excited about digital making.



To find out more, join us at

www.codeclubworld.org

Code Club is part of the Raspberry Pi Foundation. Registered Charity Number 1129409

WIN!

In association with Astroprint

ASTROBOX TOUCH & GATEWAY

WE'VE GOT FOUR ASTROBOX KITS UP FOR GRABS:

2 × AstroBox TOUCH

2 × AstroBox GATEWAY

AstroPrint makes 3D printing easy with its AstroBox Touch and Gateway kits. These devices enable you to monitor and control a 3D printer from anywhere in the world.

Most 3D printers come with very basic controls, and AstroBox uses a Raspberry Pi to add a professional software layer on top of a basic 3D printer.

With AstroBox you can download 3D printer files from a range of websites and print the models directly from your printer.

We reviewed the AstroBox Touch in issue 70 (magpi.cc/70) and found it to be "a pleasurable and intuitive experience" that turned a regular 3D printer into a very refined product.

Enter now at magpi.cc/win

Learn more: astroprint.com

Terms & Conditions

Competition opens on **27 June 2018** and closes on **27 July 2018**. Prize is offered to participants worldwide aged 13 or over, except employees of the Raspberry Pi Foundation, the prize supplier, their families or friends. Winners will be notified by email no more than 30 days after the competition closes. By entering the competition, the winner consents to any publicity generated from the competition, in print and online. Participants agree to receive occasional newsletters from The MagPi magazine. We don't like spam: participants' details will remain strictly confidential and won't be shared with third parties. Prizes are non-negotiable and no cash alternative will be offered. Winners will be contacted by email to arrange delivery. Any winners who have not responded 60 days after the initial email is sent will have their prize revoked. This promotion is in no way sponsored, endorsed or administered by, or associated with, Instagram or Facebook.

MATT RICHARDSON

Matt Richardson is the Executive Director of the Raspberry Pi Foundation North America and author of *Getting Started with Raspberry Pi*. Contact him on Twitter @**MattRichardson**.

SHOW & TELL

Matt Richardson on how sharing is a critical part of creating

Many years ago at a Maker Faire, I received a bracelet with the words 'Show what you make, share what you learn' inscribed on it. Ever since then, those words have stuck with me because they encapsulate such an important aspect of the Raspberry Pi community and the maker movement at large. We're not a community only because we have a common interest in making things with technology. We're a community because we generously share our projects and our knowledge. It's an important part of the maker ethos and it's why events like Maker Faire are such a key aspect of the movement.

Meet and share

As I've discussed here before, there's a fantastic community of Raspberry Pi enthusiasts sharing projects and knowledge online on sites such as Reddit, Twitter, Hackaday, Make, and Instructables. But nothing quite beats face-to-face interaction for sharing and community-building. We at the Raspberry Pi Foundation want to provide opportunities for members of our community to see and meet each other and have the opportunity to share.

For example, Coolest Projects is a showcase event for young people to share what they've created with technology. This event sprang from the CoderDojo movement and has become huge in Dublin over the past few years. We launched this event in the United Kingdom a few months ago and we're excited to bring it to California in September for young digital makers in North America. More details about it can be found at **coolestprojects.org/northamerica**.

For youth and adults in the Raspberry Pi community, Raspberry Fields is an opportunity to share their projects, products, and anything related to Raspberry Pi. The event is new this year and takes place in Cambridge during the summer. Read more about it in this issue of *The MagPi*. And if you aren't able to make it to Raspberry Fields this time around, check out the Raspberry Jam map (**rpf.io/jam**) for a Jam happening near you.

The purpose of events such as Coolest Projects, Raspberry Fields, Maker Faires, and Raspberry Jams is to give makers a platform to show their projects and share the knowledge they've gained through the process of making. When a person creates something or figures something out, it's natural for them to want to share that object and the experience of creating it with others. Have you ever solved a tough real-world problem and just couldn't wait to tell a friend about how you did it? Sharing that experience greatly enhances the feeling of accomplishment.

> "When makers share, it inspires, encourages, and educates others"

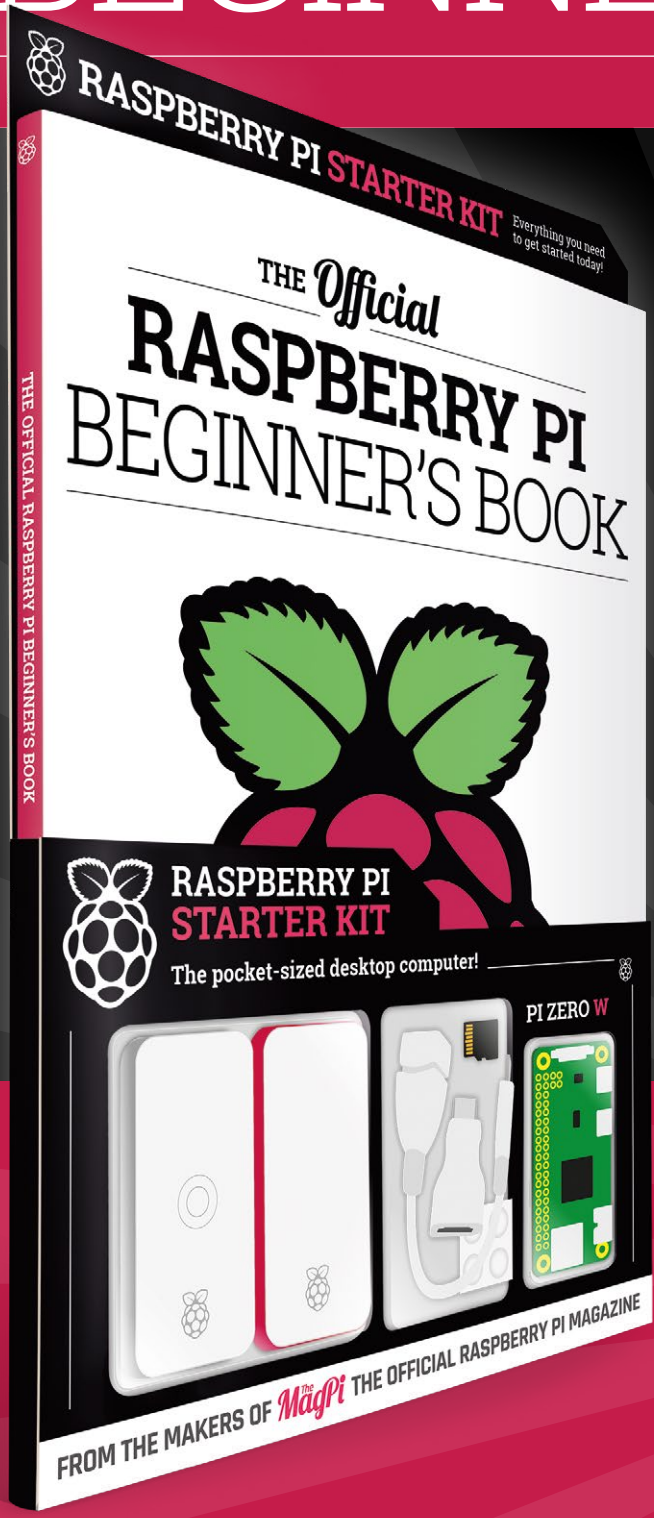
That feeling of accomplishment will give you motivation to continue creating and solving.

Inspire others

When makers share, it inspires, encourages, and educates others. I joined the maker movement after I saw the projects that others were making and I said to myself, "I wish I could do that too." Because I was able to talk to those makers and hear the details about how they created their projects, I felt supported and encouraged to try it on my own. I went from wishing I could do it to actually doing it because of how much inspiration and knowledge those makers had shared. And now I don't consider my own projects complete until I've shared them with the world.

Could you even imagine what it would be like if members of our community didn't share their projects and knowledge so freely? If that were the case, I don't even think we could consider ourselves a community. Sharing is what makes us more than just a group of people with a common interest; it is what makes us a community.

THE *Official* RASPBERRY PI BEGINNER'S BOOK


RASPBERRY PI STARTER KIT
Everything you need to get started today!
THE Official RASPBERRY PI BEGINNER'S BOOK
THE OFFICIAL RASPBERRY PI BEGINNER'S BOOK
RASPBERRY PI STARTER KIT
The pocket-sized desktop computer!
PI ZERO W
FROM THE MAKERS OF The MagPi THE OFFICIAL RASPBERRY PI MAGAZINE

LEARN COMPUTING THE EASY WAY!

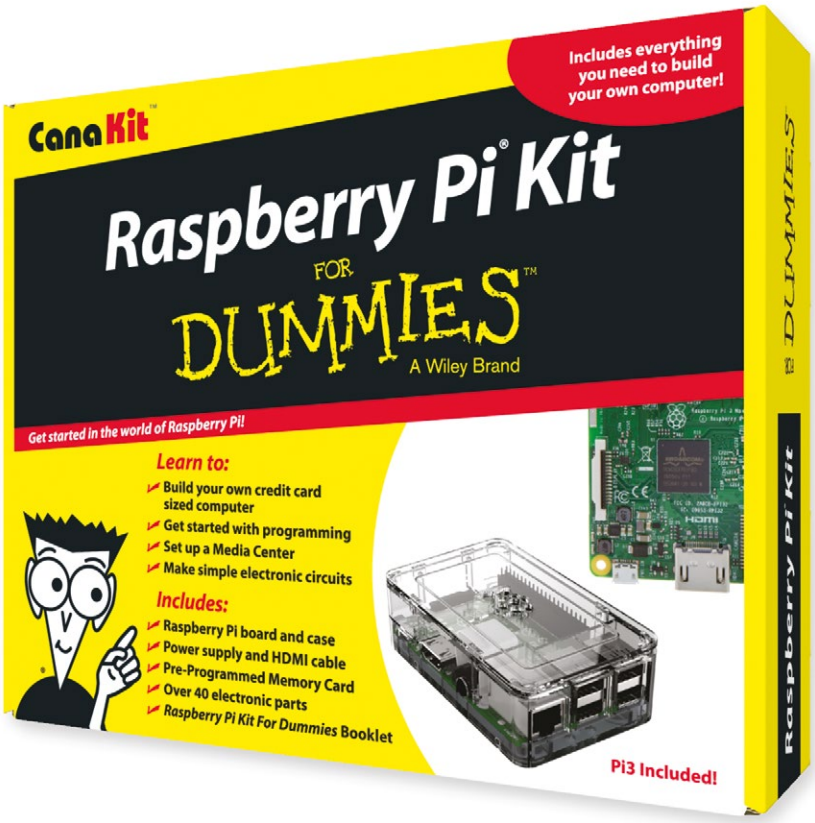
Includes

- Pi Zero W computer
- Official case with three covers
- USB and HDMI adapters
- 8GB microSD card
- 116-page beginner's book

Available now

Buy online: magpi.cc/store

CanaKit™



FOR DUMMIES® A Wiley Brand

Kit Includes:

- *Raspberry Pi For Dummies* Booklet
- Raspberry Pi 3 Board
- Memory Card
- Plastic Case
- 2.5A Power Supply
- HDMI Cable
- Resistors
- LEDs
- Push Button Switches
- Prototyping Breadboard
- Jumper Wires
- Heat Sinks

Available for worldwide shipping at:

WWW.CANAKIT.COM

Available in Europe through RS Components

$89.99 US DOLLARS

£69.99 EXCLUDING VAT

Raspberry Pi is a registered trademark of the Raspberry Pi Foundation. For Dummies and the Dummies Man logo are trademarks or registered trademarks of John Wiley & Sons, Inc. Used under license. RS logo is a registered trademark of RS Components Ltd. CanaKit is a registered trademark of Cana Kit Corporation.

ELECTRONIC KITS • ELECTRONIC PARTS • RASPBERRY PI • ARDUINO